Studienarbeit

Entwicklung eines Dashboards für das digitale Produktionsmanagement im industriellen Umfeld

Studiengang Informatik
OST - Ostschweizer Fachhochschule
Campus Rapperswil-Jona

Herbstsemester 2024

Autoren: Anian Gabathuler, Lukas Gächter
Betreuer: Prof. Dr.-Ing. Frieder Loch
Projektpartnerin: SFS Group Schweiz AG, Heerbrugg

Abstract

Die SFS Group Schweiz AG ist ein international tätiger Industriekonzern, der sich auf die Entwicklung und Herstellung von Präzisionskomponenten und Baugruppen, mechanischen Befestigungssystemen, Qualitätswerkzeugen und Bewirtschaftungslösungen spezialisiert hat. Mit rund 13.200 Mitarbeitenden weltweit hat das Unternehmen seinen Hauptsitz in Heerbrugg.

In den globalen Produktionswerken der SFS Group AG werden täglich eine Vielzahl von Kennzahlen erfasst, um die Leistung der Produktionsanlagen zu überwachen und zu optimieren. Diese Daten werden in Echtzeit erfasst und in einem zentralen System gespeichert. Zur Analyse und Überwachung der Kennzahlen kommen verschiedene Applikationen zum Einsatz.

Ziel dieser Studienarbeit ist es, die Nutzung dieser Kennzahlen durch die direkten Führungskräfte der Produktion zu fördern. Um dieses Ziel zu erreichen, sollen bestehende Applikationen erweitert oder ein Prototyp für eine neue Applikation entwickelt werden.

Zu Beginn der Arbeit wurde eine umfassende Analyse der bestehenden Kennzahlen und der genutzten Applikationen durchgeführt. Parallel dazu wurde die Nutzung der Kennzahlen durch die Führungskräfte untersucht. Im Rahmen eines ausführlichen Contextual Inquiry wurden die Anforderungen, Arbeitsabläufe und Problemstellungen der Führungskräfte an verschiedenen Produktionsstandorten ermittelt. Basierend auf diesen Erkenntnissen wurden ein Affinity-Diagramm sowie mehrere Personas erstellt. Die Personas halfen dabei, die Anforderungen der Führungskräfte zu strukturieren und zu priorisieren. Aus diesen Anforderungen wurden verschiedene Lösungsideen abgeleitet, von denen eine zur Umsetzung in einem Prototyp ausgewählt wurde.

Die Entscheidung fiel auf die Entwicklung einer Webapplikation zur digitalen Durchführung der Teammeetings. In enger Zusammenarbeit mit dem Auftraggeber wurden die Requirements definiert, um die Nutzerbedürfnisse optimal zu erfüllen. Zur strukturierten Entwicklung des Prototyps wurden mehrere Wireframes erstellt, die als Grundlage für die Implementierung dienten.

Im weiteren Verlauf der Arbeit wurde ein Prototyp entwickelt, der die digitale Durchführung von Shopfloor-Meetings ermöglicht. Ein zentrales Element der Applikation ist die Möglichkeit, personalisierte Dashboards zu erstellen, die den Führungskräften die wichtigsten Kennzahlen übersichtlich präsentieren. Der Inhalt der Dashboards kann von den Führungskräften individuell in Form von Widgets angepasst werden. Diese Widgets sind in ihrer Grösse und Anordnung flexibel, wodurch eine benutzerfreundliche und anpassbare Darstellung gewährleistet wird.

Die Evaluation des Prototyps erfolgte durch einen Usability-Test und ein abschliessendes Gespräch mit dem Auftraggeber. Dabei wurde geprüft, ob der Prototyp die definierten Anforderungen erfüllt und mit der im Vorfeld erarbeiteten Beschreibung des Minimum Viable Product (MVP) übereinstimmt. Die Ergebnisse zeigten, dass die definierten Anforderungen erfüllt wurden. Dies bestätigt den Erfolg der entwickelten Lösung und bietet eine solide Grundlage für eine mögliche Weiterentwicklung.

Management Summary

Ausgangslage

Die SFS Group Schweiz AG ist ein international tätiger Industriekonzern, der sich auf die Entwicklung und Herstellung von Präzisionskomponenten und Baugruppen, mechanischen Befestigungssystemen, Qualitätswerkzeugen und Bewirtschaftungslösungen spezialisiert hat. Mit rund 13.200 Mitarbeitenden weltweit hat das Unternehmen seinen Hauptsitz in Heerbrugg.

In den Produktionsstandorten der SFS Group AG auf der ganzen Welt werden täglich zahlreiche Informationen zur Leistung der Produktionsanlagen erfasst, um die Effizienz und Produktivität zu steigern. Diese Daten werden in Echtzeit gesammelt und zentral gespeichert. Zur Analyse und Überwachung der Informationen kommen verschiedene digitale Anwendungen zum Einsatz.

Vorgehen

Ziel dieser Arbeit war es, die Nutzung der gesammelten Informationen durch die verantwortlichen Führungskräfte in der Produktion zu fördern. Um dieses Ziel zu erreichen, wurde untersucht, wie bestehende digitale Anwendungen erweitert oder eine neue Lösung entwickelt werden könnte.

Zu Beginn der Arbeit wurde eine umfassende Analyse der bestehenden Daten und der genutzten Anwendungen durchgeführt. Gleichzeitig wurde die Nutzung der Informationen durch die Führungskräfte untersucht. Dazu wurden Gespräche mit den Beteiligten geführt, um deren Bedürfnisse, Arbeitsabläufe und Herausforderungen zu verstehen. Die daraus gewonnenen Erkenntnisse wurden strukturiert und die Bedürfnisse der Führungskräfte priorisiert. Daraus ergaben sich verschiedene Lösungsideen, von denen eine zur Umsetzung ausgewählt wurde.

Im nächsten Schritt wurde eine digitale Lösung entwickelt, die es den Führungskräften ermöglicht, ihre Besprechungen einfacher und effizienter zu organisieren. Gemeinsam mit dem Auftraggeber wurden die Anforderungen festgelegt, um sicherzustellen, dass die Bedürfnisse der Führungskräfte bestmöglich berücksichtigt werden. Als Grundlage für die Entwicklung der digitalen Lösung wurden erste Entwürfe erstellt, um die Struktur und Funktionalitäten der Anwendung zu visualisieren.

Ergebnisse

Im Verlauf der Arbeit wurde ein Prototyp einer Anwendung entwickelt, der es den Führungskräften ermöglicht, Teammeetings digital durchzuführen. Ein zentrales Element dieser Lösung ist die Möglichkeit, individuelle Übersichten der relevanten Kennzahlen zu erstellen, auf denen die wichtigsten Informationen übersichtlich dargestellt werden. Die Inhalte dieser Übersichten können von den Führungskräften selbst bestimmt und angepasst werden. Zudem lassen sich die einzelnen Elemente in diesen Übersichten flexibel anordnen und in der Grösse verändern, um eine benutzerfreundliche Darstellung sicherzustellen.

Die Anwendung wurde durch einen Benutzertest mit Führungskräften überprüft, um sicherzustellen, dass sie die definierten Anforderungen erfüllt. Die Ergebnisse des Tests zeigten, dass die Anwendung die Anforderungen erfüllt. Dies bestätigt den Erfolg der entwickelten Lösung und bietet eine solide Grundlage für eine mögliche Weiterentwicklung.

Inhaltsverzeichnis

Abkürzungsverzeichnis

API Application Programming Interface. 31, 33, 34, 36, 38

App Application. 3, 13, 14, 19

CD Continuous Deployment. 26, 36, 37

CI Continuous Integration. 26, 36, 37

Codegen Code Generation. 34

ECTS European Credit Transfer and Accumulation System. 10

HTTP Hypertext Transfer Protocol. 33

JSON JavaScript Object Notation. 33

KPI Key Performance Indicator. 18, 19

MVP Minimum Viable Product. 20, 21, 35, 37, 39

NFR Non-Functional Requirement. 20, 37

UAT User Access Terminal. 3, 19

UC Use Case. 20, 37

UI User Interface. 20

URL Uniform Resource Locator. 36

UX User Experience. 12, 13

Webapp Web Application. 27, 36

Glossar

Access Control Access Control ist eine Sicherheitstechnik, die regelt, wer oder was Ressourcen in einer Computerumgebung ansehen oder nutzen kann. 17

Accessibility Accessibility ist das Design von Produkten, Geräten, Dienstleistungen oder Umgebungen für Menschen mit Behinderungen. 17, 19

Affinity Diagram Ein Affinitätsdiagramm ist ein Werkzeug zur Organisation von Ideen und Daten. 3, 12, 16, 17, 19

Angular Angular ist eine Plattform und ein Framework zum Erstellen von Single-Page-Client-Anwendungen mit HTML und TypeScript. 29, 34

Best Practice Eine Best Practice ist eine Methode oder Technik, die konsistent bessere Ergebnisse liefert als andere Ansätze und als Benchmark verwendet wird. 3, 11, 15, 32

Branch Ein Branch ist eine parallele Version eines Repositories. 26

Card Sorting Card Sorting ist eine Technik im User Experience Design. 12

Cluster Ein Cluster ist eine Gruppe ähnlicher Elemente. 12, 16, 17, 19

Cockpit Eine Applikation, die den Führungskräften zentral alle wichtigen Kennzahlen zur Verfügung stellt. 21

Contextual Inquiry Contextual Inquiry ist eine nutzerzentrierte Designforschungsmethode und Teil der Methodik des kontextuellen Designs. 3, 11, 15

Dashboard Ein Dashboard ist eine grafische Benutzeroberfläche, die eine Echtzeitübersicht über wichtige Informationen bietet. 19–21, 27, 30, 37

Dependencies Dependencies sind externe Bibliotheken oder Module, die für die Ausführung eines Projekts erforderlich sind. 26

Feedback Feedback sind Informationen über Reaktionen auf ein Produkt, eine Leistung oder eine Aufgabe. 16

Functionality Functionality ist die Eigenschaft, einen Zweck gut zu erfüllen. 17, 19

GitLab GitLab ist ein webbasiertes DevOps-Lifecycle-Tool, das ein Git-Repository-Manager mit Wiki-, Issue-Tracking- und CI/CD-Pipeline-Funktionen bietet. 26

Header Ein Header ist ein Abschnitt eines Dokuments, der oben auf einer Seite erscheint. 30

High-Fidelity High-Fidelity bezeichnet einen Prototyp, der sehr detailliert ist. 13

Hover Hover beschreibt den Akt, den Cursor über ein Element zu bewegen. 27

Leader Bezieht sich auf eine Applikation, die nur für Führungskräfte bestimmt ist. 21

Local Storage Local Storage ist eine Methode, Daten in einem Webbrowser zu speichern. 20

Low-Fidelity Low-Fidelity bezeichnet einen Prototyp, der nicht sehr detailliert ist. 13

Management Management ist der Prozess, Dinge oder Personen zu steuern oder zu kontrollieren. 17

Mapping Mapping ist eine Technik im User Experience Design. 12

Merge Merge beschreibt die Aktion, Code aus verschiedenen Branches zusammenzuführen. 26

Miro Miro ist eine Online-Plattform für kollaborative Whiteboards. 12, 16, 17

Observables Observables sind ein neues Primitiv, das die Grundlage der reaktiven Programmierung in Angular bildet. 32

Persona Eine Persona ist eine fiktive Figur, die einen Nutzer darstellt, der eine Webseite, Marke oder ein Produkt auf ähnliche Weise nutzt. 3, 12, 18, 19

Pipeline Eine Pipeline ist eine Reihe von automatisierten Prozessen, die es ermöglichen, Code zu erstellen, zu testen und bereitzustellen. 26

Proxy Ein Proxy ist ein Zwischenserver, der Endnutzer von den Webseiten trennt, die sie besuchen. 34

Push Push beschreibt die Aktion, Code in ein entferntes Repository zu senden. 26

Remote Remote beschreibt einen Prozess, der aus der Ferne durchgeführt wird. 15

Shopfloor Ein Shopfloor ist eine grafische Benutzeroberfläche, die eine statische Übersicht über wichtige Informationen bietet. 3, 18–20, 24, 27, 30, 37, 38

Stage Eine Stage ist ein Schritt in einer Pipeline. 26

Sticky Notes Sticky Notes sind kleine Papierstücke mit einem wiederklebbaren Streifen auf der Rückseite. 12, 16, 17

Template Ein Template ist eine Datei, die als Ausgangspunkt für ein neues Dokument dient. 16, 17

Tool Ein Tool ist ein Objekt, das genutzt werden kann, um ein Ziel zu erreichen. 3, 15, 18, 19

Toolbar Eine Toolbar ist ein grafisches Bedienelement, auf dem Bildschirmtasten, Symbole, Menüs oder andere Eingabe- oder Ausgabeelemente platziert sind. 27

TypeScript TypeScript ist eine Programmiersprache, die von Microsoft entwickelt und gepflegt wird. 28

Usability Usability ist die Benutzerfreundlichkeit und Erlernbarkeit eines von Menschen geschaffenen Objekts. 17

Usability Testing Usability Testing ist eine Methode im nutzerzentrierten Interaktionsdesign, um ein Produkt durch Tests mit Nutzern zu evaluieren. 3, 11, 14

Usability-Test Usability-Tests sind eine Methode zur Bewertung eines Produkts durch Tests mit Nutzern. 11, 13, 34

User Ein User ist eine Person, die ein Produkt oder eine Dienstleistung nutzt. 21

User Experience Design User Experience Design (UXD) ist der Prozess zur Verbesserung der Nutzerzufriedenheit mit einem Produkt durch Steigerung der Benutzungsfreundlichkeit, Zugänglichkeit und Freude an der Interaktion mit dem Produkt. 11

User-Centered Design User-Centered Design ist ein iterativer Designprozess, bei dem Designer sich in jeder Phase auf die Nutzer und ihre Bedürfnisse konzentrieren. 12

Whiteboards Ein Whiteboard ist eine glänzende, normalerweise weisse Fläche für nichtpermanente Markierungen. 12

Widget Ein Widget ist ein grafisches Element auf einer Benutzeroberfläche, das frei bewegt werden kann und relevante Informationen anzeigt. 27, 30, 37, 38

Wireframe Ein Wireframe ist ein visuelles Hilfsmittel, das das Skelettgerüst einer Webseite darstellt. 3, 13, 20, 21, 27

1 Einleitung

Die Einleitung gliedert sich in drei Abschnitte: Ausgangslage, Aufgabenstellung und Rahmenbedingungen. Die Ausgangslage beschreibt die aktuellen Herausforderungen im Umgang mit den von der SFS zur Verfügung gestellten Applikationen. Anschliessend wird in der Aufgabenstellung das Ziel sowie der Lösungsansatz dieser Herausforderungen skizziert. Abschliessend werden in den Rahmenbedingungen die zeitlichen und organisatorischen Grundlagen der Arbeit erläutert.

1.1 Ausgangslage

Die SFS Group Schweiz AG ist ein international tätiger Industriekonzern, der sich auf die Entwicklung und Herstellung von Präzisionskomponenten und Baugruppen, mechanischen Befestigungssystemen, Qualitätswerkzeugen und Bewirtschaftungslösungen spezialisiert hat.

Am Standort Heerbrugg existieren unterschiedliche Applikationen, die eine Vielzahl von Kennzahlen beinhalten. In dieser Arbeit wird nach Wegen gesucht, um die Nutzung der gesammelten Informationen durch die Führungskräfte in der Produktion zu fördern. Dazu wird eine Analyse der bestehenden Kennzahlen durchgeführt. Ebenfalls wird die Nutzung der Kennzahlen durch die Führungskräfte untersucht. Das Ziel ist es, Ideen zu erarbeiten, um die Verwendung von Kennzahlen durch die Führungskräfte zu verbessern. Eine dieser Ideen wird in einem Prototypen umgesetzt und evaluiert.

1.2 Aufgabenstellung

Das vom Projektpartner genannte Hauptziel ist es, die Nutzung der Kennzahlen durch die Führungskräfte zu fördern und sicherzustellen, dass diese in die Entscheidungsprozesse einfliessen. Als Zielgruppe wurden ausdrücklich direkte Führungskräfte in der Produktion genannt.

Um dieses Ziel zu erreichen, wird eine vertiefte Analyse durchgeführt. Diese soll die bestehenden Prozesse und Systeme aufzeigen und die Anforderungen der Führungskräfte erheben. Die Analyse soll auch die technischen Rahmenbedingungen und die bestehenden Datenquellen berücksichtigen.

Basierend auf dieser Analyse sollen Ideen erarbeitet werden, um die Nutzung der Kennzahlen zu verbessern. Eine dieser Ideen wird in einem Prototyp umgesetzt und evaluiert. Dabei soll ein menschzentrierter Ansatz verfolgt werden.

Abschliessend findet eine kritische Reflektion statt und es werden konkrete Verbesserungsvorschläge diskutiert. Das ganze Projekt wird durch eine geeignete Projektmanagementmethode unterstützt. Die Arbeit wird durch Arbeitspakete und Meilensteine strukturiert und die Planung regelmässig überprüft.

1.3 Rahmenbedingungen

Diese Arbeit wird im Rahmen einer Studienarbeit durchgeführt. Das Zeitbudget beträgt pro Person 240 Stunden, was in der Gesamtheit 480 Stunden ergibt. Die Vergütung erfolgt in Form von 8 ECTS-Punkten.

2 Literaturrecherche

Im Rahmen dieser Arbeit wurden verschiedene Methoden und Techniken aus dem Bereich des User Experience Design und des Usability Testing analysiert. Ziel war es, bewährte Praktiken zu identifizieren, die bei der Evaluation der Anforderungen, Entwicklung des Prototyps und der Durchführung von Usability-Tests hilfreich sein können. Die folgenden Abschnitte geben einen Überblick über die wichtigsten Erkenntnisse und Best Practices, die im Rahmen der Literaturrecherche gewonnen wurden.

2.1 Contextual Inquiry

Das Contextual Inquiry ist eine qualitative Forschungsmethode, die darauf abzielt, das Verhalten und die Arbeitsprozesse von Nutzenden durch Beobachtung und Befragung in ihrer natürlichen Umgebung zu verstehen. Entwickelt von Hugh Beyer und Karen Holtzblatt, behebt diese Methode die Schwächen anderer qualitativer Methoden wie Umfragen und Interviews, die oft auf das Erinnerungsvermögen der Nutzenden angewiesen sind. Beim Contextual Inquiry werden Nutzende direkt während der Ausführung von Aufgaben beobachtet, was es den Forschenden ermöglicht, tiefere Einblicke in die tatsächlichen Prozesse, Motivationen und mentalen Modelle der Nutzenden zu gewinnen.

Diese Methode ist besonders wertvoll in den frühen Phasen der Produktentwicklung. Durch die direkte Beobachtung der Nutzenden können Forschende unvorhergesehene Details, Unterbrechungen und gewohnheitsmässige Verhaltensweisen erkennen, die die Designentscheidungen beeinflussen können. Es gibt jedoch auch Risiken, wie dass Nutzende in den Interviewmodus wechseln oder dass die Forschenden unbewusst ihre eigenen Vorurteile einbringen, was die Ergebnisse verfälschen kann. [1]

Die Methode folgt vier zentralen Prinzipien:

- **Kontext**: Forschende beobachten Nutzende in ihrer tatsächlichen Arbeitsumgebung.
- **Partnerschaft**: Forschende und Nutzende arbeiten zusammen, um den Arbeitsprozess zu verstehen.
- **Interpretation**: Forschende interpretieren das Verhalten der Nutzenden mit deren Rückmeldungen.
- **Fokus**: Die Forschung orientiert sich an spezifischen Zielen, die sich auf das Produktdesign beziehen.

Eine typische Sitzung ist in vier Phasen unterteilt:

1. **Einleitung**: Aufbau einer Beziehung und Erhebung von Vorabinformationen.
2. **Übergang**: Wechsel in die Beobachtungsphase, bei der die Rolle der Forschenden erklärt wird.
3. **Kontextuelles Interview**: Beobachten, Lernen und Nachfragen bei Unklarheiten.
4. **Abschluss**: Klärung der Erkenntnisse und Zusammenfassung der Prozesse für ein finales Feedback.

2.2 Affinity Diagram

Affinity Diagramming, oft auch als Affinity Mapping bezeichnet, ist eine Methode zur Gruppierung von zusammenhängenden Beobachtungen, Ideen, Konzepten oder Erkenntnissen in klar definierte Cluster. In Abgrenzung zum Card Sorting konzentriert sich das Affinity Diagramming dabei eher auf das Zusammenfassen breiterer Ideen. Es wird häufig in UX-Workshops eingesetzt, um Designideen zu entwickeln und konkrete nächste Schritte im Projekt zu definieren.

2.2.1 Anwendungsbereiche im UX-Bereich

Im UX-Prozess ist das Affinity Diagramming eine flexible Mapping-Aktivität, die in verschiedenen Phasen eingesetzt werden kann, von der anfänglichen Forschung bis zur finalen Designoptimierung. Sie ermöglicht es Einzelpersonen und Teams, Ideen aus der Forschung, Konzepte aus dem Design sowie strategische Visionen zu ordnen und gemeinsam zu diskutieren.

2.2.2 Ablauf

Affinity Diagramming folgt meistens einem dreistufigen Prozess:

- **Ideengenerierung mittels Sticky Notes**: Einzelpersonen notieren Ideen auf Sticky Notes und verteilen sie ungeordnet. So werden unterschiedliche Perspektiven erfasst, ohne dass einzelne Stimmen dominieren.
- **Organisation der Notizen in Cluster oder Themen**: Nach der Ideengenerierung werden die Sticky Notes in thematische Cluster gruppiert. Hier entstehen erste Oberkategorien.
- **Priorisierung der Cluster und Festlegung von nächsten Schritten**: Am Ende werden Cluster priorisiert, um wichtige Themen zu fokussieren und konkrete Massnahmen zu formulieren.

2.2.3 Praxis-Tipps

Für ein effektives Affinity Diagramming sollten grosse Datenmengen im Vorfeld auf Relevanz geprüft werden, damit Sitzungen nicht überladen wirken. Verschiedene Perspektiven sollten berücksichtigt werden, um eine ausgewogene Diskussion zu fördern. Es ist wichtig, dass Cluster eindeutig beschriftet und nicht gezwungen zusammengefasst werden, da eine klare Struktur die spätere Arbeit mit dem Diagramm vereinfacht.

2.2.4 Fazit

Das Affinity Diagramming ist eine wertvolle kollaborative Methode im UX-Design, die komplexe Informationen in handhabbare Handlungsschritte überführt. Während der Prozess des Diagramms oft wichtiger ist als das Ergebnis selbst, trägt die gemeinsame Erarbeitung dazu bei, das Team zu vereinen und nächste Schritte zu definieren. Diese Methode eignet sich besonders für digitale Whiteboards wie Miro, wo Ideen schnell dokumentiert und organisiert werden können. [3]

2.3 Personas

Personas sind von grosser Bedeutung als methodisches Werkzeug im User-Centered Design. Ziel ist es, die Zielgruppe eines Produkts durch fiktive, jedoch auf Daten basierende Charaktere zu beschreiben. Dadurch sollen spezifische Bedürfnisse der Nutzenden im Designprozess berücksichtigt und kommuniziert werden.

2.3.1 Definition

Personas sind detaillierte Modelle, die typische Verhaltensmuster, Ziele und Bedürfnisse einer Gruppe von Nutzenden abbilden. Sie dienen als eine Art Steckbrief einer fiktiven Person, die wesentliche Merkmale der Zielgruppe zusammenfasst, wie Aufgaben, Ziele, Ängste und Nutzungskontexte. Diese Persona-Steckbriefe ermöglichen eine präzisere und emotional ansprechende Darstellung der Bedürfnisse.

2.3.2 Erstellung

Die Entwicklung von Personas basiert auf qualitativen Analysen und Methoden wie Feldstudien, Interviews und Umfragen. Die im Rahmen dieser Analysen gewonnenen Daten werden zu realistischen fiktiven Personen gebündelt, die stellvertretend für typische Gruppen von Nutzenden stehen. Idealerweise wird die Persona-Erstellung im frühen Stadium eines Projekts im Team durchgeführt, um eine stärkere Bindung der Beteiligten an die Daten der Nutzenden zu fördern und die Akzeptanz für die Nutzung von Personas zu steigern.

2.3.3 Vorteile

Personas helfen, Designentscheidungen gezielt und konsistent an den realen Anforderungen der Zielgruppe auszurichten. In Team-Meetings bieten sie eine einheitliche und spezifische Referenz für Diskussionen, die dabei unterstützt, auf Nutzende ausgerichtete Funktionalitäten zu priorisieren. Das Design wird so klar auf die definierten Zielgruppenbedürfnisse fokussiert und verhindert eine allzu breite Streuung auf mögliche Randfälle.

2.3.4 Langfristige Anwendung

Auch über die Designphase hinaus bleiben Personas wertvoll. Sie können als Vorlage für Usability-Tests dienen, bei der Segmentierung von Analysedaten unterstützen und bei der Zusammenarbeit mit externen Agenturen eine klare Zielgruppenbeschreibung liefern. Da sich die Bedürfnisse von Nutzenden im Laufe der Zeit ändern können, sollten Personas kontinuierlich anhand neuer Daten aktualisiert und als lebendige Dokumente betrachtet werden, die regelmässig verfeinert werden. [6]

2.4 Wireframes

Wireframing ist eine Methode, um das Grundgerüst von Websites und Apps zu erstellen und die Führung von Nutzenden sowie die Informationsarchitektur frühzeitig zu planen. Mit Wireframes können die Struktur einer Seite, die Platzierung von Elementen und die Benutzerfreundlichkeit bereits in der Konzeptphase visualisiert werden. Diese Technik hilft, UX-Probleme frühzeitig zu erkennen und ermöglicht eine auf die Nutzenden ausgelegte Gestaltung.

Ein Wireframe stellt das Basisdesign als visuelles Diagramm dar und dient als Grundlage für detailliertere Entwürfe. In der UX-Entwicklung ist ein Low-Fidelity-Wireframe der erste Schritt und bietet eine grobe Skizze der Struktur. High-Fidelity-Wireframes erweitern dies um interaktive Elemente und Details und eignen sich für erste Tests mit Nutzenden. [4]

2.4.1 Typen von Wireframes

- **App- und Website-Wireframes**: Visualisieren die Interaktionen und Benutzeroberflächen und helfen, die Bedienbarkeit und Struktur zu optimieren.
- **Produktentwicklung**: Produktverantwortliche nutzen Wireframes, um die Logik, Funktionalität und Priorisierung in der Produktentwicklung zu skizzieren.
- **Projektmanagement**: Im Projektmanagement dienen Wireframes der Klarstellung von

Zweck, Ziel und geplanten Ergebnissen eines Projekts und fördern eine abgestimmte Kommunikation.

2.5 Usability Testing

Usability Testing ist ein zentraler Bestandteil der Entwicklung benutzerfreundlicher und intuitiver Websites und Anwendungen. Dieser Abschnitt beleuchtet zwei wichtige Aspekte des Usability Testings: unmoderierte Usability-Tests und Website Usability Testing.

2.5.1 Unmoderierte Usability-Tests

Unmoderierte Usability-Tests sind eine effiziente Methode, um Feedback einzuholen, ohne dass jede Sitzung persönlich begleitet werden muss.

Die Hauptvorteile unmoderierter Tests sind die Geschwindigkeit und Skalierbarkeit. Da keine individuellen Meetings mit den Teilnehmenden geplant werden müssen, können Studien in wenigen Stunden durchgeführt werden. Allerdings gibt es Einschränkungen:

- **Eingeschränkte Interaktion:** Ohne Moderation fehlt die Möglichkeit, auf spezifische Fragen einzugehen oder die Teilnehmenden bei der Navigation durch komplexe Prototypen zu unterstützen.
- **Geringere Motivation:** Teilnehmende verhalten sich möglicherweise weniger engagiert oder realistisch, insbesondere bei Aufgaben, die Vorstellungskraft oder emotionale Reaktionen erfordern.

Unmoderierte Tests eignen sich besonders für die Evaluation von funktionalen Prototypen sowie bestehenden Websites und Apps, bei denen realistische Interaktionen analysiert werden sollen. [6]

2.5.2 Website Usability Testing

Website Usability Testing fokussiert sich auf die Benutzerfreundlichkeit und Intuitivität einer Website. Ziel ist es, sicherzustellen, dass die Nutzenden reibungslos mit der Website interagieren können. Dies ist entscheidend, da ein Grossteil der Nutzenden eine Website nach zwei schlechten Erfahrungen verlassen. [8]

2.5.3 Vorteile

Die Durchführung solcher Tests bringt zahlreiche Vorteile:

- Verbesserung der Benutzerzufriedenheit und Erhöhung der Bindungsrate.
- Optimierung von Prozessen.
- Aufbau von Vertrauen und Glaubwürdigkeit durch eine funktionale und ansprechende Website.
- Erfüllung verschiedener Standards, insbesondere im Bereich Barrierefreiheit.

2.5.4 Ablauf

Ein typischer Prozess umfasst folgende Schritte:

1. **Ziele definieren:** Festlegen, was mit dem Test erreicht werden soll.

2. **Zielgruppe identifizieren:** Ermittlung der Bedürfnisse von Nutzenden, Schmerzpunkte und Verhaltensweisen.
3. **Testmethoden auswählen:** Entscheiden zwischen moderierten, unmoderierten, Remote oder vor Ort Tests.
4. **Testszenarien erstellen:** Realistische Aufgaben, die die Zielsetzung des Tests widerspiegeln.
5. **Testumgebung einrichten:** Auswahl geeigneter Tools wie plattformübergreifende Testsysteme.
6. **Pilot-Test durchführen:** Erste Tests im kleinen Rahmen, um Fehler in den Anweisungen zu erkennen.
7. **Testsitzung durchführen und überwachen:** Aufgaben beobachten und Problempunkte analysieren.
8. **Daten sammeln:** Erhebung qualitativer und quantitativer Daten.
9. **Ergebnisse analysieren:** Muster und Herausforderungen identifizieren und priorisieren.
10. **Erneut testen:** Änderungen validieren, indem der Testprozess wiederholt wird.
11. **Dokumentation:** Erstellung eines Berichts mit den wichtigsten Erkenntnissen und Anpassungen.

2.5.5 Best Practices

Für effektives Usability Testing sollten folgende Best Practices berücksichtigt werden:

- Nutzende auswählen, die nicht direkt in das Projekt involviert sind, um eine neutrale Perspektive zu gewährleisten.
- Aufgaben offen formulieren, um unvorhergesehene Hindernisse und Usability-Probleme zu identifizieren.
- Usability-Tests in verschiedenen Entwicklungsphasen wiederholen, um kontinuierlich Verbesserungen einzubringen.

2.6 Contextual Inquiry

2.6.1 Vorbereitung

Um das Verbesserungspotential und die Anforderungen an die Software zu ermitteln, werden mehrere Contextual Inquiries durchgeführt. Im Folgenden wird von Teamleitern gesprochen. Diese Bezeichnung bezieht sich auf die sechs Personen, die Teil der Contextual Inquiries sind. In den übrigen Teilen der Arbeit wird von Führungskräften gesprochen, da damit alle Personen gemeint sind, die eine leitende Funktion innehaben und Teil der Zielgruppe sind.

Für die Durchführung der Contexutal Inquiries haben wir einen strukturierten Leitfaden erstellt. Dieser dient als roter Faden für die Contextual Inquiries und gewährleistet, dass alle relevanten Themen systematisch angesprochen werden. Der Leitfaden enthält nicht nur einen Ablaufplan, sondern auch eine Liste der wichtigsten Themen, die während der Contextual Inquiries adressiert werden sollen. Zu diesen Themen gehören:

- Die alltäglichen Aufgaben und Arbeitsprozesse der Teamleiter

- Die Zufriedenheit der Teamleiter mit der aktuellen Software
- Die Herausforderungen und Schwierigkeiten, die die Teamleiter bei ihrer Arbeit haben
- Mögliche Wünsche und Verbesserungsvorschläge für die bestehenden Applikationen

Zusätzlich zum Leitfaden haben wir ein strukturiertes Template erstellt, das dazu dient, die Antworten der Teamleiter während der Contextual Inquiries festzuhalten. Das Template enthält nicht nur die wichtigsten Angaben zur befragten Person, sondern auch eine Reihe von allgemeineren Fragen. Diese Fragen zielen darauf ab, die individuelle Arbeitsweise der Teamleiter zu verstehen sowie herauszufinden, wie die aktuelle Software in den jeweiligen Arbeitsablauf integriert ist. Das ermöglicht eine Vergleichbarkeit der Daten und erleichtert die anschliessende Auswertung und Analyse.

Im Rahmen der Contextual Inquiries konzentrieren wir uns auf die Zielgruppe der Teamleiter und deren Arbeitsweise. Um ein breites und differenziertes Bild ihrer Arbeitsweise zu erhalten, haben wir Termine mit insgesamt sechs Teamleitern vereinbart. Die ausgewählten Teamleiter stammen aus verschiedenen Abteilungen und Standorten. Jedes Contextual Inquiry dauert ungefähr eine Stunde, wobei die Teamleiter ihre täglichen Arbeitsprozesse ausführen und beschreiben. Die Contextual Inquiries werden von zwei Personen durchgeführt, wobei eine Person die Rolle des Beobachters einnimmt und die Fragen stellt, während sich die andere Person auf das Festhalten der Antworten konzentriert.

2.6.2 Durchführung

Für die Contextual Inquries wurden sechs Teamleiter aus verschiedenen Abteilungen und Standorten beobachtet. Die Contextual Inquiries fanden dabei in der täglichen Arbeitsumgebung der Teamleiter statt, während diese ihre typischen Arbeitsprozesse ausführten.

Zu Beginn der Contextual Inquiries wurde uns jeweils der allgemeine Ablauf der täglichen Arbeit eines Teamleiters nähergebracht. Es wurde gezeigt, wie die Teamleiter ihre Aufgaben organisieren, welche Software sie dafür verwenden und auf welche Schwierigkeiten sie dabei stossen. Ebenfalls wurden Wünsche und Verbesserungsvorschläge zur Software thematisiert. Offene Fragen und Unklarheiten konnten im Laufe der Inquiries direkt mit den Teamleitern besprochen werden, was zu einem tiefergehenden Verständnis der Arbeitsprozesse führte.

Im Rahmen des Contextual Inquiries haben wir den Teamleitern bei ihren alltäglichen Aufgaben über die Schulter geschaut. Dadurch konnten die Teamleiter ihre Arbeitsweise und den Umgang mit der Software erläutern, während wir spezifische Fragen stellten und direktes Feedback erhielten.

Durch die Contextual Inquiries konnten wir eine breite Datenbasis sammeln, die uns dabei hilft, die Anforderungen und Bedürfnisse der Teamleiter besser zu verstehen und einzuordnen.

2.6.3 Auswertung

Als Datenbasis für die Auswertung der Contextual Inquiries dienten die ausgefüllten Templates mit den Notizen und Antworten aus den Inquries. Für die Auswertung selbst haben wir uns für das Affinity Diagramming entschieden, da diese Methode eine effiziente und strukturierte Möglichkeit bietet, um die gesammelten Daten zu analysieren und zu kategorisieren. [3] Hierfür wurde Miro mit einem entsprechenden Template verwendet, um die Daten zu strukturieren und zu visualisieren.

Zu Beginn wurden die Daten in Miro auf Sticky Notes übertragen und anschliessend in thematische Cluster gruppiert. Diese Cluster wurden dann benannt und priorisiert, um die wichtigsten Themen und Anforderungen zu identifizieren.

2.7 Affinity Diagram

In einem ersten Schritt wurden die Daten aus den Contextual Inquiries in einem entsprechenden Template von Miro auf Sticky Notes übertragen. Dabei haben wir die Daten farblich nach dem Teil des Inquiries sortiert, aus dem sie hervorgegangen sind. Die drei Hauptkategorien hierbei sind die Daten zu den Arbeitsabläufen, zum allgemeinen Eindruck und der Nutzungshäufigkeit sowie zu den Wünschen und Verbesserungsvorschlägen.

Anschliessend haben wir die Sticky Notes zu Clustern zusammengefasst, die thematisch zusammengehören. Jeder dieser Cluster wurde mit einem Namen versehen, der den Inhalt des Clusters zusammenfasst. Nach diesem Schritt war die Anzahl der Cluster jedoch noch zu gross. Deshalb haben wir die Namen der Cluster als Ausgangspunkt genommen, um die Cluster noch weiter zusammenzufassen. Daraus ergaben sich fünf finale Cluster. Zu diesen gehören Access Control, Management, Accessibility, Functionality und Usability, wobei Functionality und Usability deutlich umfangreicher sind als die anderen drei Cluster.

In einem nächsten Schritt wurden für jeden der finalen Cluster Ideen formuliert, wie die Probleme, die in den Clustern angesprochen wurden, gelöst werden könnten. Zu diesen Ideen haben wir auch Fragen formuliert, die sich bei diesen Ideen stellen und die in den folgenden Schritten beantwortet werden sollen. Danach wurde jedem der finalen Cluster eine Priorität zugewiesen. Diese basieren auf der Häufigkeit der Problemnennung während der Inquiries, der Eignung der Ideen als Thema für die Studienarbeit sowie der Anzahl der Ideen, die zu jedem Cluster formuliert wurden.

Zum Schluss haben wir die erarbeiteten Prioritäten in einer Priorisierungsliste als Rangliste zusammengefasst. Aus dieser Liste konnte abgeleitet werden, auf welche der finalen Cluster und deren Ideen wir uns konzentrieren möchten, um daraus umfangreiche Ideen für mögliche Themen der Studienarbeit zu formulieren. Dabei lag der Fokus auf den Clustern Functionality, Management und Accessibility, da diese die höchsten Prioritäten aufweisen und somit die grössten Verbesserungspotentiale bieten.

2.8 Personas

Basierend auf den Daten aus den Contextual Inquiries haben wir drei Personas erstellt, die die wichtigsten Typen von Führungskräften repräsentieren, welche mit dem Tool arbeiten werden. Bei der Erstellung der Personas wurde darauf geachtet, dass sie so realistisch wie möglich sind und die Bedürfnisse und Ziele der Führungskräfte widerspiegeln. Die Bilder zu den Personas wurden über eine Webseite generiert, die künstliche Personenbilder erstellt. [13]

Die erste Persona repräsentiert eine pragmatische Führungskraft. Sie kommt mit den Tools zurecht, verwendet sie aber nur für das Nötigste. Ihr Fokus liegt auf den KPIs und der Effizienz des Teams. Sie möchte so wenig Zeit wie möglich mit den Tools verbringen und sich auf die Arbeit konzentrieren.

Die zweite Persona repräsentiert eine Führungskraft, die sich auf die Datenoptimierung fokussiert. Sie hat sehr gute Excel-Kenntnisse und hat sogar eine eigene Excel-Lösung für die Shopfloor-Daten aus den verschiedenen Tools entworfen. Sie möchte die Visualisierung der Daten selbst festlegen, um die Daten so verständlich wie möglich zu präsentieren. Zudem ermöglicht ihr die Excel-Lösung eine benutzerdefinierte Auswertung der Daten.

Die dritte Persona repräsentiert eine teamzentrierte und unterstützende Führungskraft. Ihre Arbeitsweise orientiert sich stark an den Bedürfnissen des Teams. Sie möchte das Team in seiner Arbeit unterstützen und ihm die bestmöglichen Bedingungen bieten. Dadurch erhofft sie sich eine Steigerung der Effizienz und der Motivation des Teams.

2.9 Evaluation Themenidee

Basierend auf den Ideen aus dem Affinity Diagram und den erstellten Personas haben wir zwei Ideen für mögliche Themen ausformuliert.

2.9.1 Thema 1: Shopfloor-Tool für Führungskräfte

Diese Idee fokussiert sich auf die Ergebnisse aus dem Functionality Cluster des Affinity Diagrams sowie auf die Datenoptimierer-Persona. Die Idee ist es, ein Tool zu entwickeln, das es Führungskräften ermöglicht, die Daten aus den verschiedenen Tools zu visualisieren und zu personalisieren. Die Führungskräfte sollen die Möglichkeit haben, eigene Dashboards für ihre Shopfloors zu erstellen und die Daten so zu visualisieren, wie sie es für richtig halten. Das soll dazu führen, dass die Führungskräfte ihre Shopfloors einheitlicher abhalten, sich mehr an die vorgegebenen KPIs halten und keine Zeitverluste durch Eigenlösungen mehr haben.

2.9.2 Thema 2: Tool oder App für das UAT

Diese Idee fokussiert sich auf die Ergebnisse aus dem Accessibility Cluster des Affinity Diagrams sowie auf die teamorientierte Unterstützer-Persona. Die Idee ist es, ein Tool oder eine App zu entwickeln, die eine abgespeckte Version des UATs darstellt. Das soll dazu beitragen, dass die Mitarbeitenden an den Anlagen relevante Dokumente und Hilfestellungen rund um die Anlage jederzeit einsehen können. Dadurch sollen die Mitarbeitenden besser auf die Anlagen vorbereitet sein und die Anlagen effizienter bedienen können.

2.9.3 Themenauswahl

Die von uns ausformulierten Themenideen haben wir dem Auftraggeber präsentiert und mit ihm die Vor- und Nachteile der beiden Themen diskutiert. Am Ende dieses Meetings fiel die Entscheidung auf Thema 1, da es das Kernziel dieser Studienarbeit besser abbildet und einen direkteren Effekt auf die Führungskräfte hat.

3 Konzept

In diesem Kapitel wird das Konzept des MVPs für die Führungskräfte im Shopfloor vorgestellt. Es beschreibt die Erstellung und Anpassung von Dashboards sowie die Verwendung dieser als Grundlage für Shopfloor-Übersichten. Zudem werden die Anforderungen des MVPs und die entsprechenden Wireframes präsentiert, die das Design und die Struktur der Anwendung visualisieren. Dabei werden auch Anpassungen zur Verbesserung der Benutzerfreundlichkeit erläutert.

3.1 MVP-Definition

Die Führungskräfte haben die Möglichkeit personalisierbare Dashboards als Vorlage für ihre Shopfloors zu erstellen. Dazu können sie aus vordefinierten Widgets auswählen und diese frei auf der Seite platzieren und ihre Grösse anpassen. Änderungen an den personalisierten Dashboards werden automatisch übernommen. Die Führungskräfte können für den jeweils aktuellen Shopfloor auf Basis der personalisierten Dashboards eine neue Shopfloor-Übersicht erstellen. Die erstellten Shopfloor-Übersichten können nicht mehr bearbeitet werden, sondern dienen als eine Art Snapshot des zugrunde liegenden Dashboards und werden zur Präsentation der relevanten Daten verwendet.

Beim MVP werden die erstellen Dashboards und Shopfloor-Übersichten im Local Storage des Browsers gespeichert. Das UI des MVP orientiert sich am User- und Leader-Cockpit sowie den Guidelines für Dashboards, um für die Führungskräfte möglichst intuitiv bedienbar zu sein. Der MVP wird modular aufgebaut, um die Erweiterbarkeit mit weiteren Widgets und zusätzlichen Funktionalitäten einfach zu ermöglichen.

Für den MVP sind basierend auf obenstehender Beschreibung die folgenden Anforderungen relevant:

- **UC1**: Erstellung eines personalisierten Dashboards
- **UC4**: Historienverwaltung der Shopfloor-Pages
- **NFR1**: Intuitive Benutzeroberfläche
- **NFR2**: Modularer Aufbau für Erweiterbarkeit
- **NFR6**: Lauffähiger Main-Branch

3.2 Wireframes

Basierend auf den Anforderungen und dem MVP haben wir Wireframes erstellt, um die Struktur und das Design der Applikation zu visualisieren. Dabei lag der Fokus auf den wichtigsten Seiten und Funktionen, um die Umsetzung des MVP zu erleichtern. Das Design und die Farbgebung der Wireframes orientieren sich am User- und Leader-Cockpit, um eine konsistente Benutzererfahrung zu gewährleisten und den Führungskräften den Einstieg in die Applikation zu erleichtern.

Nach der Implementierung eines ersten lauffähigen Standes gab es einige Schwächen im Design der Applikation. Deshalb haben wir uns nochmals vertieft mit ähnlichen Applikationen auseinandergesetzt und zusätzliche Guidelines zu Rate gezogen. Basierend auf den daraus folgenden Erkenntnissen wurden die Wireframes überarbeitet und die Applikation angepasst. Dadurch konnten wir die Benutzerfreundlichkeit und das Design der Applikation verbessern. In den nachfolgenden Absätzen werden die überarbeiteten Wireframes für die wichtigsten Seiten und Funktionen der Applikation erläutert.

3.2.1 Auswahl der Widgets

Das erste Wireframe zeigt die Auswahl der Widgets auf dem personalisierbaren Dashboard. Auf dieser Seite können die Führungskräfte über das Symbol rechts unten ein Fester öffnen, in dem die verfügbaren Widgets grafisch dargestellt werden. Zudem bietet die Seite einige festgelegte Filter, um nur Widgets einer bestimmten Kategorie anzuzeigen. Die Widgets selbst können über ein Plus-Symbol markiert werden, damit sie später zum Dashboard hinzugefügt werden. Da das Auswahlfenster das Dashboard teilweise verdeckt, erscheint beim Hinzufügen eine Meldung unten in der Mitte des Bildschirms. Ausserdem wird der Counter in der rechten unteren Ecke des Widgets um eins erhöht. Dieser Counter zeigt an, wie oft das Widget zum Dashboard hinzugefügt werden soll.

Sobald die Führungskraft alle gewünschten Widgets markiert hat, kann sie das Auswahlfenster mit einem Klick auf das Hinzufügen-Symbol links unten schliessen. Daraufhin werden die markierten Widgets zum Dashboard hinzugefügt und die Seite wird aktualisiert.

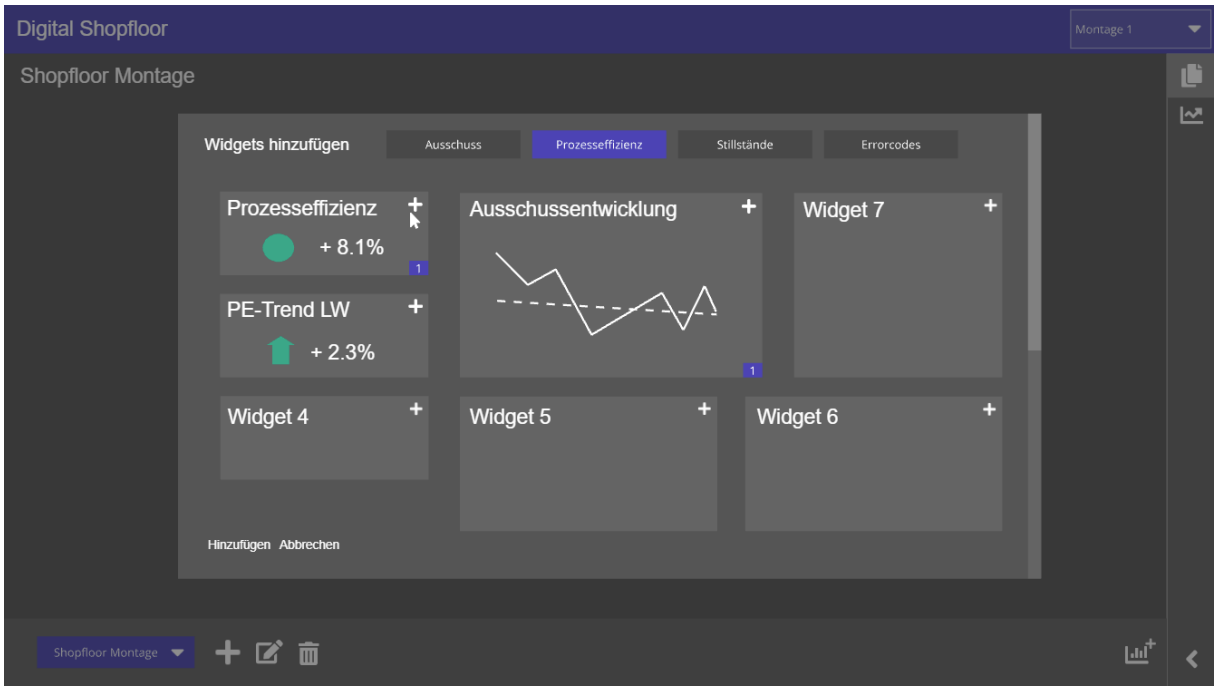


Abbildung 3.1: Widgets für Dashboard auswählen

3.2.2 Dashboard-Übersicht

Das zweite Wireframe zeigt die Auswahl des Dashboards, welches angezeigt werden soll. Dazu kann die Führungskraft unten links das gewünschte Dashboard aus einem Drop-Down-Menü auswählen. Zudem kann sie auf dieser Seite mit einem Klick auf das Plus-Symbol links unten ein neues Dashboard erstellen. Ausserdem kann das aktuell ausgewählte Dashboard mit einem Klick auf das Stift-Symbol unten links bearbeitet werden. Sollte ein Dashboard nicht mehr benötigt werden, kann die Führungskraft diese mit einem Klick auf das Mülleimer-Symbol links unten löschen.

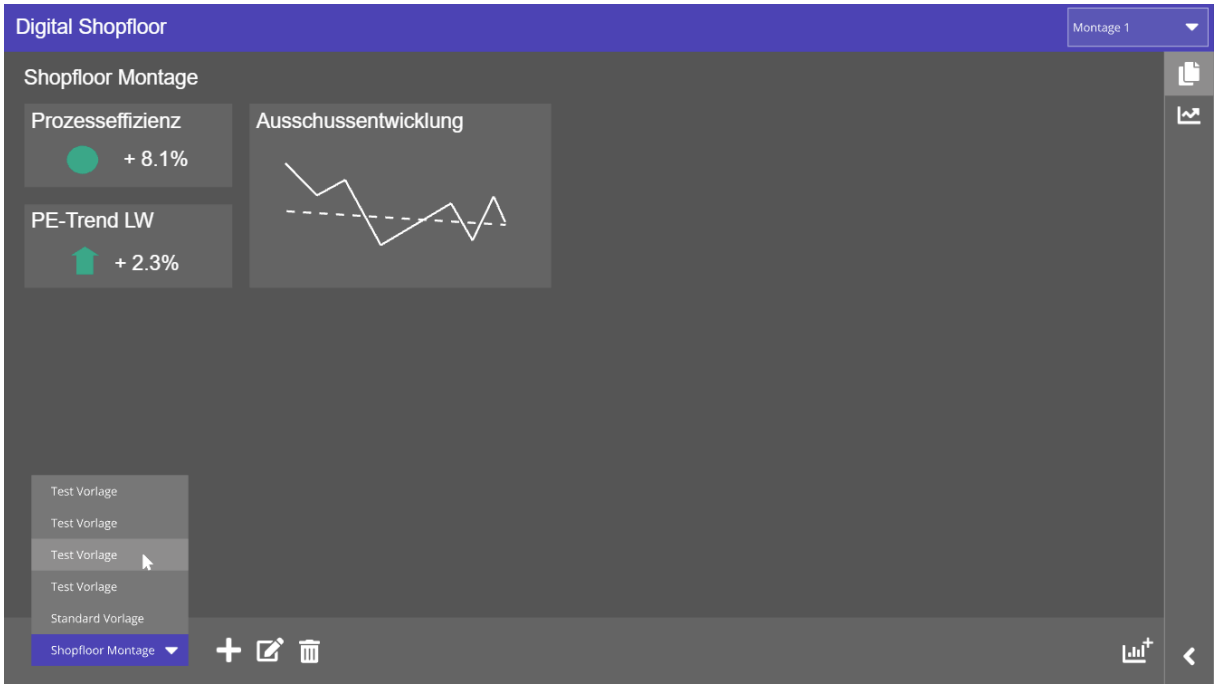


Abbildung 3.2: Dashboard auswählen

3.2.3 Konfiguration der Widgets

Das dritte Wireframe zeigt die Konfiguration der Widgets auf dem personalisierbaren Dashboard. Die Widgets können auf der Seite frei verschoben und in ihrer Grösse angepasst werden.

Zudem können die Daten, die das Widget anzeigen soll, konfiguriert werden. Dazu kann die Führungskraft in der oberen rechten Ecke eines Widgets auf das Zahnrad-Symbol klicken, um das Konfigurationsfenster zu öffnen. In diesem Fenster werden der Führungskraft alle Konfigurationsmöglichkeiten angezeigt, die für das jeweilige Widget verfügbar sind. In dem abgebildeten Wireframe handelt es sich dabei um den Zeitraum der Daten, die das Widget anzeigen soll. Dabei kann die Führungskraft aus Tagen, Wochen, Monaten und Jahren auswählen. Die Anzahl der Tage, Wochen, Monate oder Jahre kann die Führungskraft über ein Eingabefeld frei definieren. Zudem kann sie über einen Slider auswählen, ob der Zeitraum gleitend sein soll oder nicht. Wählt die Führungskraft beispielsweise als Zeitraum die letzten zwei Wochen aus und aktiviert den gleitenden Zeitraum, so zeigt das Widget die Daten der letzten zwei Wochen ausgehen vom aktuellen Datum an. Deaktiviert sie den gleitenden Zeitraum, so zeigt das Widget die Daten der letzten zwei abgeschlossenen Kalenderwochen vor dem aktuellen Datum an. Um Verwirrung vorzubeugen, wird der Führungskraft eine entsprechende Erklärung angezeigt, wenn sie mit der Maus über den Slider fährt.

Ist die Konfiguration abgeschlossen, kann das Fenster mit einem Klick auf das Speichern-Symbol unten links geschlossen werden und das Widget zeigt die konfigurierten Daten an. Soll die Konfiguration abgebrochen werden, kann das Fenster mit einem Klick auf das Abbrechen-Symbol unten links geschlossen werden.

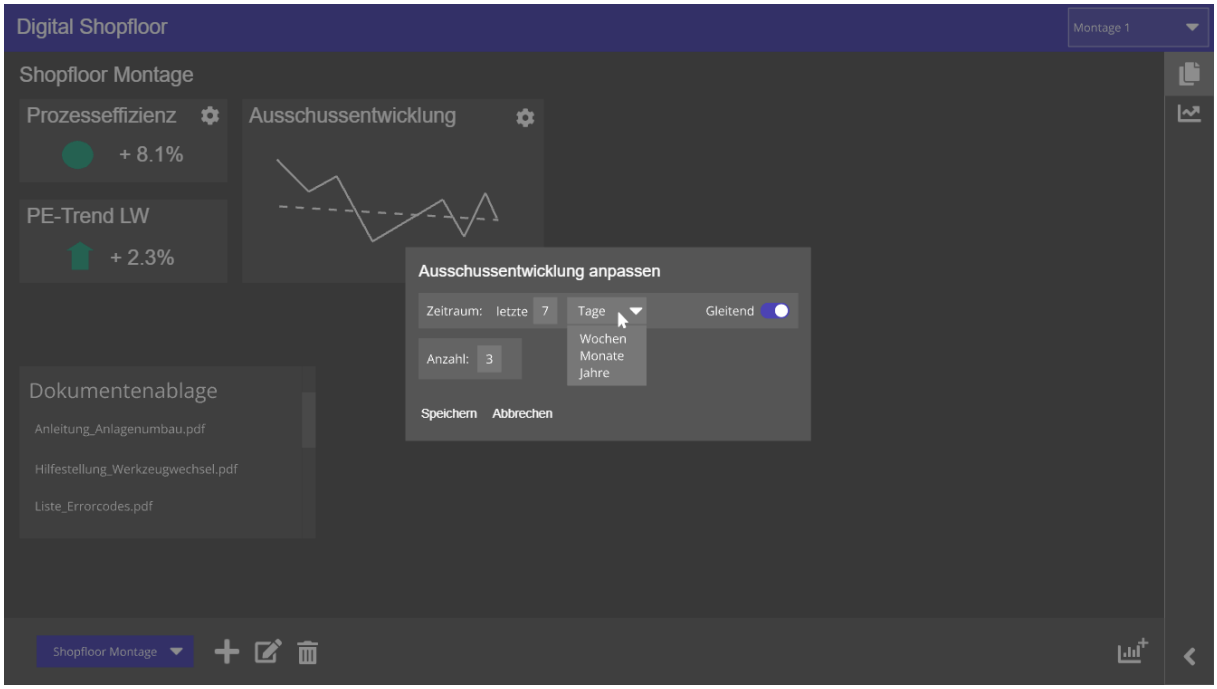


Abbildung 3.3: Widgets auf dem Dashboard konfigurieren

3.2.4 Erstellung eines Shopfloors

Das vierte Wireframe zeigt die Erstellung einer neuen Shopfloor-Page auf Basis der personalisierten Dashboards. Dazu klickt die Führungskraft auf das Plus-Symbol links unten auf der Seite. Daraufhin öffnet sich ein Fenster, indem sie zuerst das Dashboard für die neue Shopfloor-Page auswählen kann. Dabei kann sie zwischen den personalisierten Dashboards, die sie bereits erstellt hat, auswählen. Als zweiten Schritt gibt die Führungskraft das gewünschte Datum für die neue Shopfloor-Page ein oder wählt es über das Kalender-Symbol aus einem interaktiven Kalender aus. Diese Information wird benötigt, um die Daten abhängig vom eingegebenen Datum in die Widgets abzufüllen. Hat die Führungskraft beide Felder ausgefüllt, kann sie die Erstellung der neuen Shopfloor-Page über den Button Erstellen abschliessen. Sollte sie die Erstellung abbrechen wollen, kann sie dies über den Button Abbrechen tun.

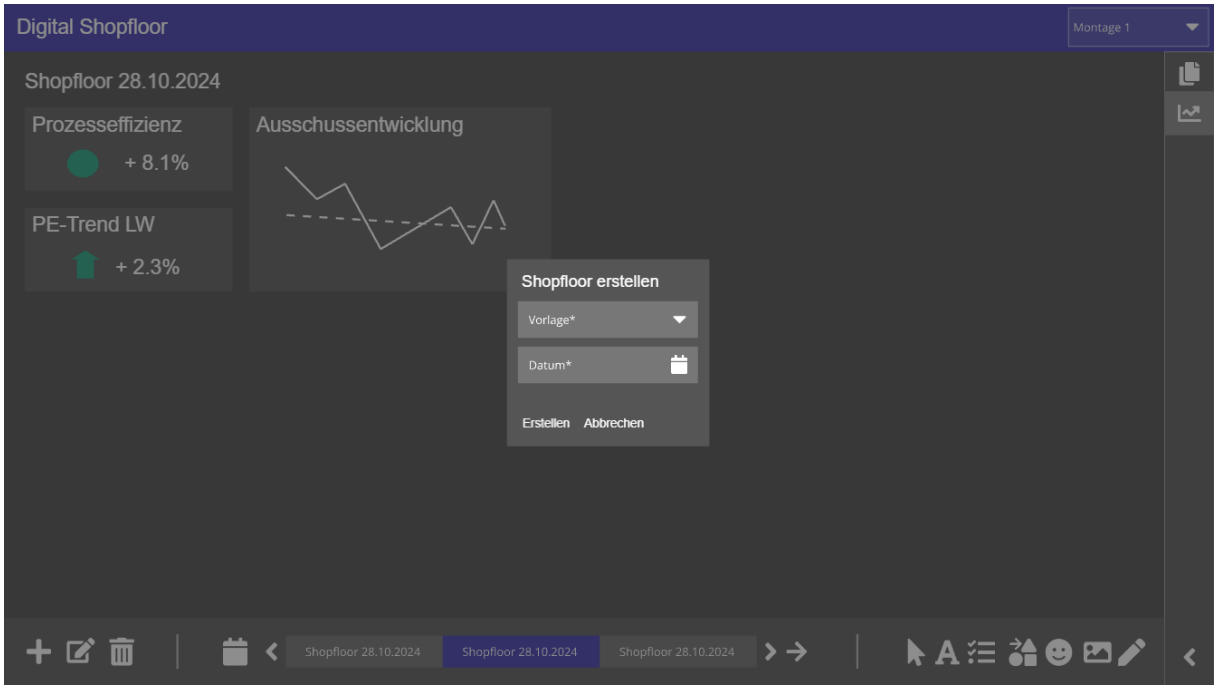


Abbildung 3.4: Erstellung einer neuen Shopfloor-Page

3.2.5 Shopfloor-Übersicht

Das fünfte Wireframe zeigt, wie eine Führungskraft den Shopfloor für die Präsentation erweitern kann. Dazu kann sie die Menüpunkte unten rechts benutzen. Über diese Menüpunkte kann sie unter anderem Textfelder, Formen und Listen zur Seite hinzufügen. Die hinzugefügten Elemente können frei verschoben und in ihrer Grösse angepasst werden. Zudem kann hier die Shopfloor-Page ausgewählt werden, welche angezeigt werden soll. Dazu kann die Führungskraft unten in der Mitte durch den Zeitstrahl navigieren oder das Kalender-Symbol benutzen, um ein bestimmtes Datum auszuwählen. Shopfloor-Pages können auf dieser Seite über die entsprechenden Symbole unten links auch hinzugefügt oder gelöscht werden. Ausserdem sieht man auf dieser Seite auch die Menüleiste auf der rechten Seite, über welche man zwischen den Vorlagen und den Shopfloors wechseln kann.

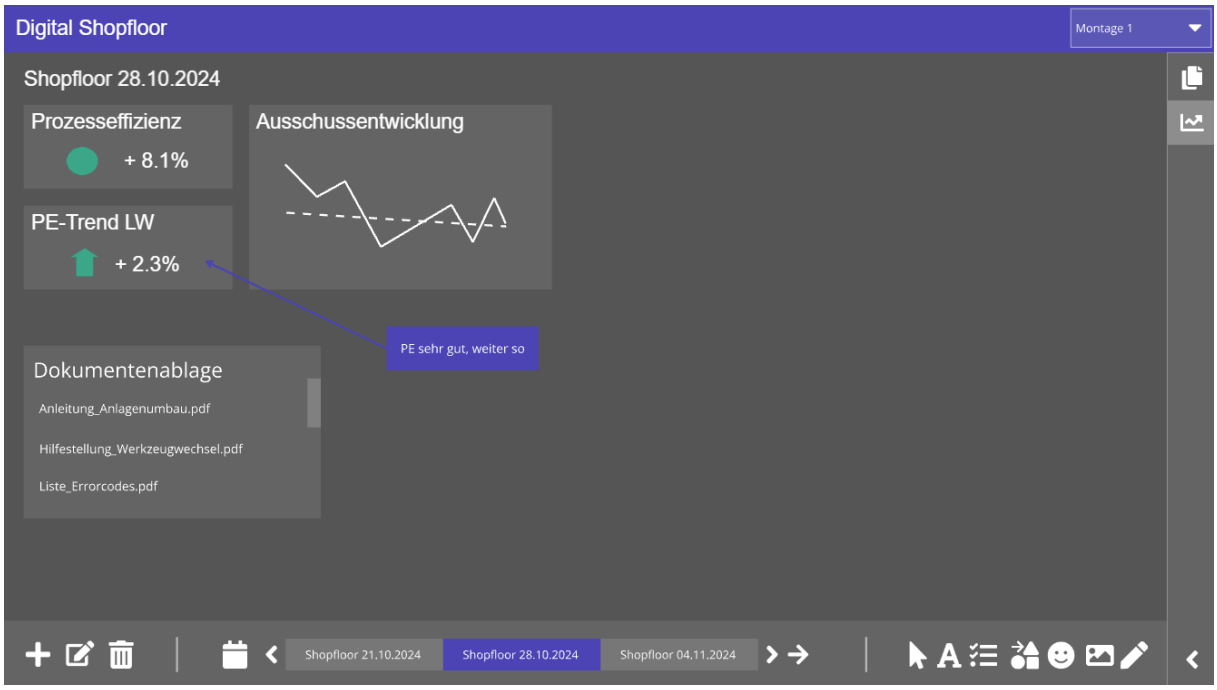


Abbildung 3.5: Shopfloor-Page für Präsentation erweitern

4 Umsetzung

Dieses Kapitel beschreibt die technische Realisierung der Applikation. Es umfasst den Aufbau der notwendigen Infrastruktur, die Entwicklung der Benutzeroberfläche sowie die Integration von Backend-Diensten. Zudem wird die Struktur der Komponenten erläutert und die Teststrategie zur Sicherstellung der Softwarequalität vorgestellt.

4.1 Infrastruktur

Unseren Code haben wir in einem GitLab-Repository abgelegt. Um die Lauffähigkeit unserer Applikation sicherzustellen, habe wir im Gitlab eine CI/CD-Pipeline eingerichtet, die bei jedem Push auf einen Branch ausgeführt wird. Der Main-Branch ist protected. Merges sind nur möglich, wenn die Pipeline erfolgreich durchgelaufen ist. Dies stellt sicher, dass nur funktionierender Code in den Main-Branch gemerged wird.

4.1.1 CI/CD-Pipeline

Die CI/CD-Pipeline besteht aus den Stages Install, Test, Build, Lint und Deploy. Im Default-Block wird das Cypress Image für alle Jobs festgelegt. Dieses Image hat den Vorteil, dass es bereits einen Browser enthält, wodurch die Tests ohne zusätzliche Installationen direkt ausgeführt werden können.

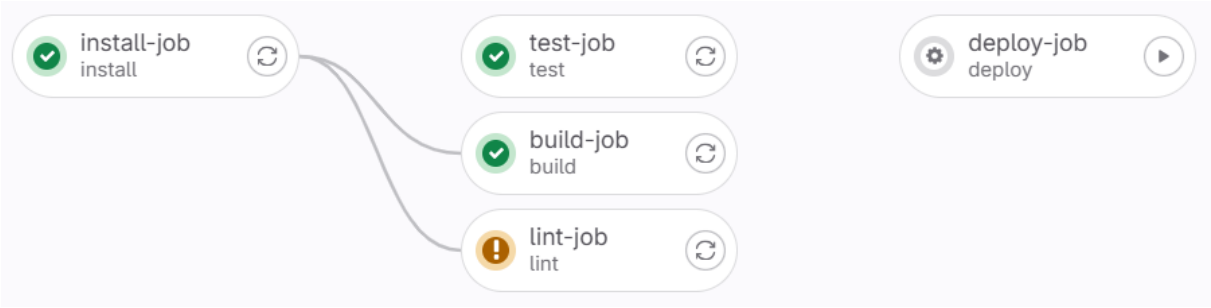


Abbildung 4.1: Ablauf und Abhängigkeiten der CI/CD Pipeline

Install

Die Install-Stage enthält den `install-job`, welcher die Dependencies installiert und den Ordner `node_modules` für die anderen Jobs bereitstellt. Dieser Job läuft zuerst, da die anderen Jobs auf die Dependencies angewiesen sind. Alle weiteren Jobs laufen parallel, da sie unabhängig voneinander sind.

Test

Die Test-Stage enthält den `test-job`, welcher sämtliche Tests aus den vorhandenen Spezifikationsfiles der Codebase ausführt. Dabei wird das Cypress Image verwendet, um die Tests automatisiert im Browser ausführen zu können.

Build

Die Build-Stage enthält den `build-job`, welcher den Code in eine produktionsfähige Version umwandelt. Allfällige Build-Fehler werden hier abgefangen.

Lint

Die Lint-Stage enthält den `lint-job`, welcher den Code auf Fehler überprüft. Warnung des Linters werden erlaubt, da diese nicht zwingend behoben werden müssen. Bei Fehlern schlägt der Job jedoch fehl. So wird das Team darauf aufmerksam gemacht, dass der Code noch überarbeitet werden muss.

Deploy

Die Deploy-Stage enthält den `deploy-job`, welcher den Code in eine öffentlich erreichbare statische Azure Webapp deployed. Der Job muss manuell gestartet werden und wurde nur verwendet, um die Applikation temporär für die Usablity-Tests bereitzustellen. Das manuelle Starten ermöglicht es, dass nur ausgewählte Versionen der Applikation deployed werden. Der Job verwendet einige Variablen wie den von der statischen Webapp generierten Bereitstellungstoken und angepasste Pfade für die Azure Webapp.

4.2 Umsetzung

Die Umsetzung des Prototyps entspricht weitestgehend den Wireframes. Die Applikation enthält je eine Seite für Dashboards und Shopfloors, die über die Navigationsleiste erreichbar sind.

Die Dashboard-Seite ermöglicht das Hinzufügen, Bearbeiten und Löschen von Dashboards. Ebenso können über einen Button in der Toolbar neue Widgets hinzugefügt werden. Das Wechseln zwischen den Dashboards erfolgt über die Dropdown-Liste in der Toolbar. Die Widgets können in dieser Ansicht per Drag and Drop frei in einem Raster platziert sowie in der Grösse angepasst werden. Das Löschen von Widgets funktioniert über einen Button in der oberen rechten Ecke des Widgets, der beim Hover angezeigt wird.

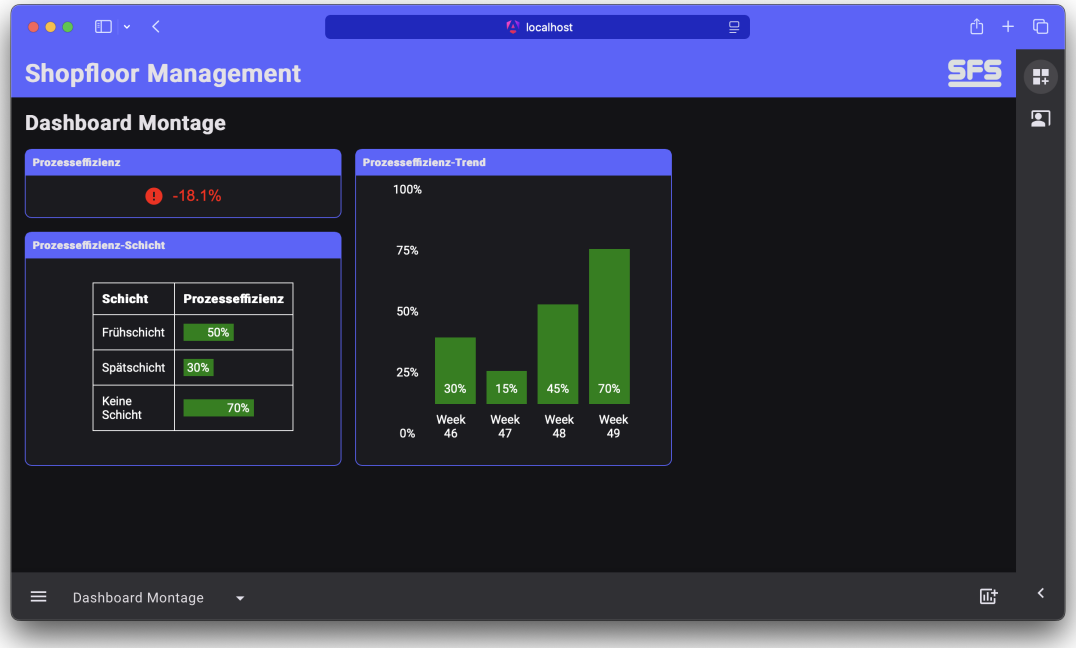


Abbildung 4.2: Dashboard mit Widgets

Die Shopfloor-Seite bietet ähnliche Funktionen wie die Dashboard-Seite. Beim Erstellen eines Shopfloors muss jedoch ein Datum und ein zuvor erstelltes Dashboard ausgewählt werden, von dem die Widgets kopiert werden. Die Widgets können in dieser Ansicht nicht mehr bearbeitet werden. Dies stellt sicher, dass die Historie der Shopfloors konsistent und nachverfolgbar bleibt. In der Toolbar gibt es eine benutzerfreundliche Selektion, mit der die nach Datum geordneten Shpfloors traversiert werden können.

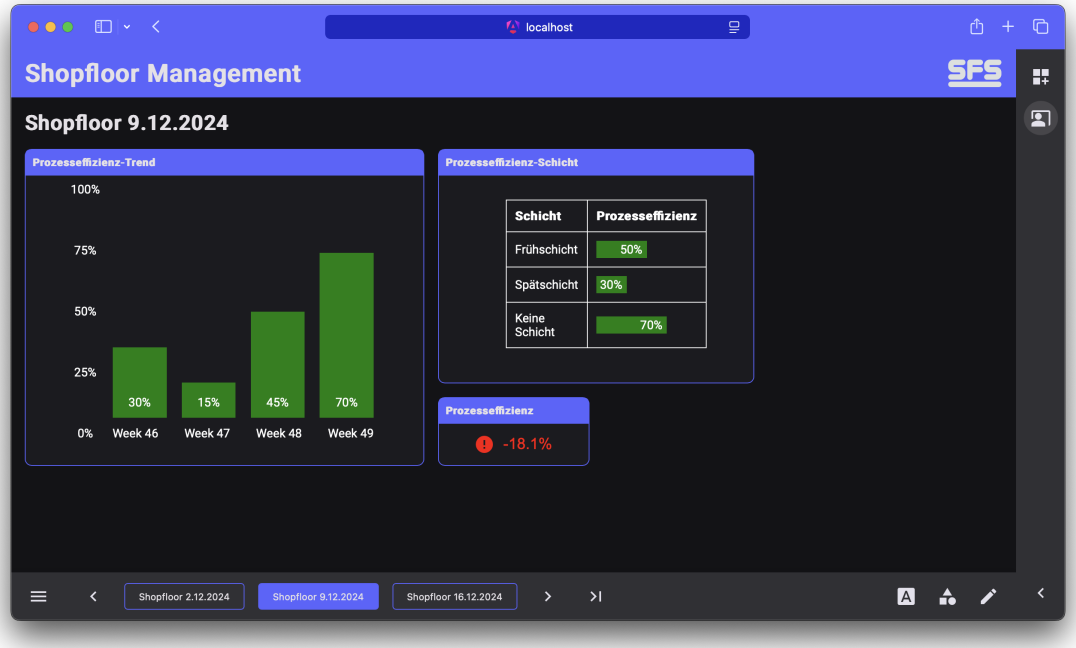


Abbildung 4.3: Shopfloor mit Widgets

4.2.1 Bibliotheken

Für die Implementation der Anwendung haben wir wie vom Auftraggeber vorgegeben Angular verwendet. Angular ist ein modernes Web-Framework, das auf TypeScript basiert und eine umfangreiche Sammlung von Bibliotheken und Tools bietet, um Webanwendungen zu entwickeln. Um Abhängigkeiten gering zu halten, haben wir nur wenige zusätzliche Bibliotheken verwendet, die für die Entwicklung der Anwendung essenziell waren. Diese sind in der nachfolgenden Tabelle aufgelistet.

Material	Die Angular Material Library bietet vorgefertigte Designkomponenten, die das Layout und die Benutzeroberfläche der Anwendung vereinfachen. Die Nutzung von Angular Material ermöglicht eine konsistente Benutzeroberfläche und eine leichtere Integration in bestehende Anwendungen des Auftraggebers.
Gridstack	Gridstack.js ist eine Bibliothek zur Erstellung dynamischer, interaktiver Dashboards. Wir haben Gridstack.js verwendet, um Widgets innerhalb der Anwendung per Drag-and-Drop zu verschieben und deren Grösse flexibel anzupassen. Die Verwendung von Gridstack.js ermöglicht eine benutzerfreundliche Oberfläche zur individuellen Anpassung von Dashboards.
RxJS	RxJS ist eine Bibliothek zur Unterstützung von ereignisbasierter und asynchroner Programmierung. RxJS wurde integriert, um asynchrone Datenströme zu verarbeiten. Durch die Nutzung von Observables können Datenänderungen in Echtzeit verfolgt werden. Dies erleichtert die Synchronisation der Benutzeroberfläche mit dynamischen Datenquellen.
Zone	Zone.js ist eine Bibliothek, die die Erkennung und Verwaltung von asynchronen Aufgaben innerhalb von Angular ermöglicht. Zone.js wird verwendet, um die Änderungsüberprüfung (Change Detection) in Angular zu automatisieren. Dadurch wird sichergestellt, dass Änderungen an Variablen oder Modellen automatisch in der Benutzeroberfläche sichtbar werden.
UUID	UUID ist eine Bibliothek zur Generierung eindeutiger Identifikationsnummern. Wir nutzen UUID, um eindeutige Kennungen für Objekte wie Dashboards oder Widgets zu generieren. Dies stellt sicher, dass keine Konflikte durch doppelte IDs in der Anwendung entstehen.
Jasmine	Jasmine ist ein Framework zum Schreiben von Unit-Tests. Wir nutzen Jasmine, um sicherzustellen, dass die einzelnen Funktionen und Komponenten wie erwartet funktionieren. Es ermöglicht die einfache Erstellung von Tests mit `describe` und `it` Blöcken, um erwartete Ergebnisse zu definieren.
Karma	Karma ist ein Test-Runner, der das parallele Ausführen von Tests in mehreren Browsern ermöglicht. Wir verwenden Karma, um die Jasmine-Tests auszuführen und die Testabdeckung zu überprüfen. Karma ermöglicht eine direkte Rückmeldung bei Änderungen des Codes und beschleunigt den Entwicklungsprozess.
ESLint	ESLint ist ein statisches Codeanalyse-Tool, das den Code auf Fehler und Stilverstösse überprüft. Wir verwenden ESLint, um den Code-Stil konsistent zu halten und potenzielle Fehler frühzeitig zu erkennen. ESLint hilft dabei, die Codequalität zu verbessern und die Wartbarkeit des Codes sicherzustellen.

Tabelle 4.1: Eingesetzte Bibliotheken

4.2.2 Komponenten

Das Frontend der Applikation ist modular aufgebaut und in verschiedene Hauptkomponenten gegliedert, die sich sowohl funktional als auch architektonisch klar voneinander abgrenzen lassen. Die Struktur spiegelt sich in der Ordnerhierarchie des Projekts wider und ermöglicht eine saubere Trennung von Zuständigkeiten. Die Komponenten sind meist in einem Container- und einem Präsentationskomponenten-Pattern implementiert, um die Logik von der Darstellung zu trennen und die Wiederverwendbarkeit zu erhöhen. Nachfolgend werden beide zur Vereinfachung zusammengefasst als einzelne Komponente betrachtet. Die folgende Abbildung zeigt eine Übersicht der zentralen Komponenten.

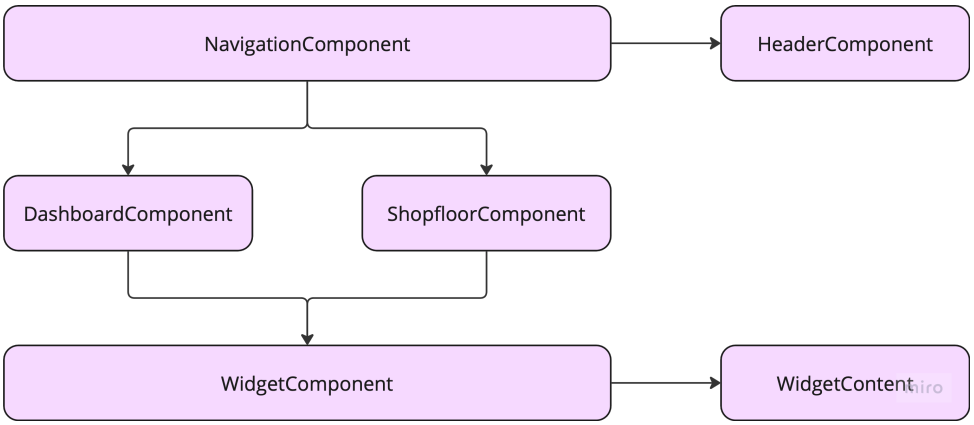


Abbildung 4.4: Hauptkomponenten der App

App

Die App-Komponente dient als Einstiegspunkt und Container für die gesamte Applikation. Sie enthält den Router, um die verschiedenen Seiten der Anwendung zu verwalten. Die ausgewählte Seite wird dynamisch an die Navigation-Komponente übergeben und darin gerendert.

Navigation

Die Navigationskomponente enthält die Seitenleiste und den Header, die die Navigation und den globalen Zustand der Anwendung steuern. Diese beiden Komponenten sind auf allen Seiten sichtbar. Die Navigation-Komponente verwendet den Router, um die Seiten zu wechseln und die aktuelle Seite zu aktualisieren.

Dashboard- und Shopfloors

Die Dashboard- und Shopfloor-Komponenten bieten ähnliche Funktionalitäten, wie das Hinzufügen, Bearbeiten und Löschen von Dashboards oder Shopfloors. Die Komponente benutzt dafür den entsprechenden Service und ist nur für die Darstellung und Weiterleitung der Benutzereingaben an den Service verantwortlich. In beiden Komponenten wird jeweils eine Toolbar bereitgestellt, die eine Interaktion mit den von der Komponente verwalteten Elementen ermöglicht. In der Toolbar der Dashboards können beispielsweise neue Dashboards erstellt oder Widgets hinzugefügt werden.

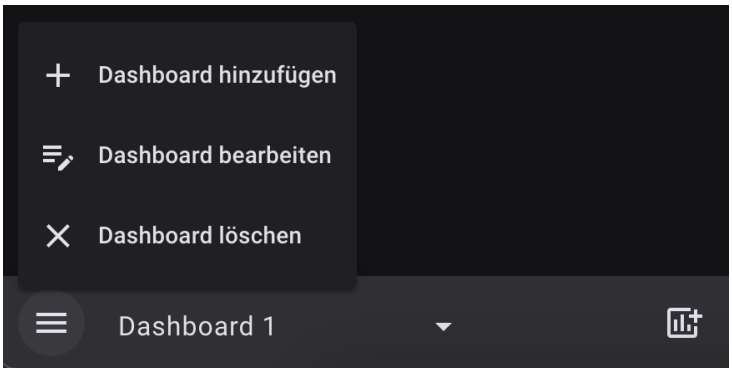


Abbildung 4.5: Toolbar der Dashboards

Widgets

Die Applikation enthält eine Basiskomponente für Widgets, die die allgemeine Struktur und das Verhalten definiert. Ausserdem implementiert sie ein GridStack-Raster, das eine freie Anordnung von GridStack-Elementen per Drag and Drop ermöglicht. GridStack-Elemente sind Container, die sich im GridStack-Raster positionieren lassen und in ihrer Grösse anpassbar sind. Ändert sich ein GridStack-Element, emittiert das GridStack-Raster ein Event, das von der Basiskomponente abgefangen wird, um die Änderungen auf dem Widget-Objekt zu speichern. Die Änderungen werden im Widget-Service gespeichert und können bei Bedarf wiederhergestellt werden.

Spezifische Widgets, die in eigenen Unterordnern organisiert sind, erweitern diese Basis-Komponente und bieten individuelle Darstellungen und Funktionen. Sie können beispielsweise Daten aus der API abrufen und in einer Tabelle oder einem Diagramm darstellen. Dafür muss der Typ, der Name und die dazugehörige Komponente im Widget-Service registriert werden. Beim Erstellen eines neuen Widgets muss der Typ ausgewählt werden und die entsprechende Komponente wird als Content in das GridStack-Element gerendert. Für jedes Widget, dass zum aktuell ausgewählten Dashboard oder Shopfloor gehört, wird eine Instanz dieser Komponente in einem GridStack-Element erzeugt.

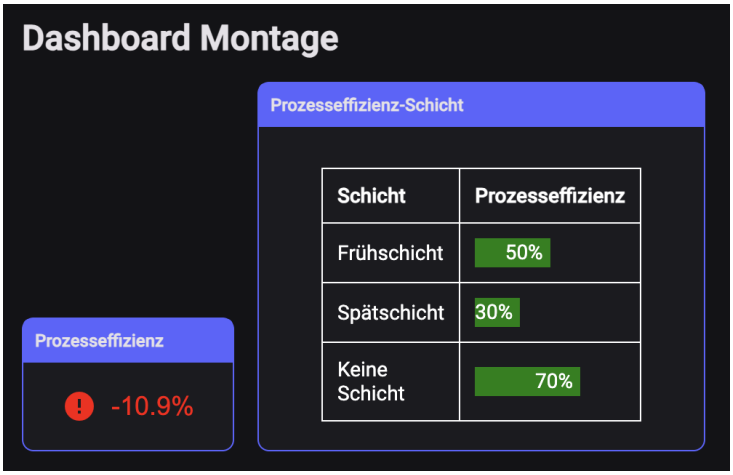


Abbildung 4.6: Zwei Widgets auf einem Dashboard

Core

Im Core ist die Implementierung der API-Schnittstellen enthalten. Sie definiert Data Transfer Objects (DTOs), um die Datenstruktur klar zu typisieren und die Kommunikation mit der Backend-API zu standardisieren.

4.2.3 Services

Wir haben uns an die Best Practices gehalten und die Logik weitestgehend in den Services implementiert. Ausserdem wird die State-Verwaltung durch diese spezialisierten Services realisiert, die als zentrale Schnittstellen fungieren. Alle Services implementieren Observables, um State-Änderungen effizient und reaktiv zu propagieren. Dies entspricht modernen Best Practices in Angular und sorgt für eine performante und wartbare Anwendung. Die nachfolgende Abbildung illustriert die wichtigsten Services der Anwendung.

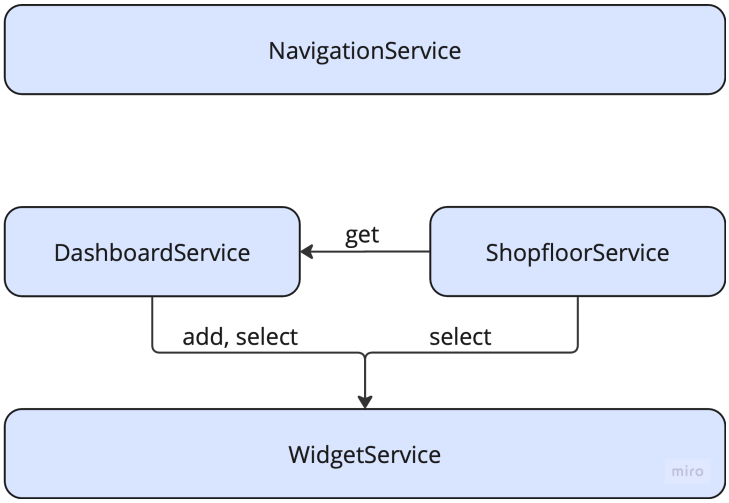


Abbildung 4.7: Services der App

Dashboard- und Shopfloors

Die Dashboard- und Shopfloor-Service implementieren die CRUD-Methoden (Create, Read, Update, Delete) für Dashboard- beziehungsweise Shopfloor-Objekte. Zudem Verwalten die Services die Selektion der ausgewählten Objekte, die von der entsprechenden Komponente anschliessend dargestellt wird. Beide Services stellen alle verfügbaren und das aktuell selektierte Objekt als Observable zur Verfügung. Abhängig von der Selektion werden immer auch die entsprechenden Widgets per Widget-Service selektiert. Die Widgets eines Shopfloors werden initial aus einem Dashboard kopiert und können danach nicht mehr bearbeitet werden, wodurch eine Kopplung zwischen diesen beiden Services entsteht.

Widgets

Der Widget-Service implementiert die CRUD-Operationen für Widgets und die Verwaltung der Widget-Selektion. Im Service sind ausserdem alle spezifischen Widgets registriert, die als Content gerendert werden können.

4.2.4 API-Integration

Die Daten, die in den Widgets angezeigt werden, stammen von einer externen API, die vom Auftraggeber bereitgestellt wurde. Diese API stellt die notwendigen Informationen im JSON-Format zur Verfügung, welche genutzt werden, um die entsprechenden Widgets dynamisch zu befüllen.

OpenAPI-Definition

Die API folgt einer OpenAPI-Spezifikation, die eine detaillierte Beschreibung der Endpunkte, Parameter und Rückgabewerte enthält. Die OpenAPI-Definition ermöglicht es, die API-Struktur klar zu verstehen und dient als Grundlage für die automatisierte Code-Generierung und Dokumentation.

API-Endpunkte

Die wichtigsten API-Endpunkte, die in unserem Projekt verwendet wurden, sind unten abgebildet:

- **/v1/process-efficiency/overview/team**: Liefert die nach Team gefilterte Prozess-Effizienz in einem definierten Zeitraum.
- **/v1/process-efficiency/overview/plant**: Liefert die nach Standorten gefilterte Prozess-Effizienz in einem definierten Zeitraum.
- **/v1/process-efficiency/overview/equipment**: Liefert die nach Anlage gefilterte Prozess-Effizienz im definierten Zeitraum.
- **/v1/master-data/teams**: Liefert die Liste aller Teams.
- **/v1/master-data/plants**: Liefert die Liste aller Standorte.
- **/v1/master-data/equipments**: Liefert die Liste von Anlage.

Verwendete Methoden und Parameter

Die API verwendet hauptsächlich die HTTP-Methode `GET`, um Daten aus den verschiedenen Endpunkten zu beziehen. Alle Anfragen beinhalten Parameter wie `startDate`, `endDate`, `groupByInterval` und optionale Filterparameter wie `filterEquipments`, `filterTeams` und `filterPlants`.

Die wichtigsten Parameter sind:

- **startDate, endDate**: Geben den Zeitraum an, für den die Prozess-Effizienz abgerufen werden soll.
- **groupByInterval**: Bestimmt das Intervall, nach dem die Effizienz gruppiert wird (Tag, Woche, Monat).
- **filterEquipments, filterTeams, filterPlants**: Optionale Filter, um die Daten nach bestimmten Anlagen, Teams oder Standorten einzugrenzen.

Integration in das Angular-Frontend

Die API wurde in das Angular-Frontend integriert, indem wir HTTP-Anfragen an die jeweiligen Endpunkte gesendet und die erhaltenen Daten in den Widgets verarbeitet haben. Zur Vereinfachung der Integration und der Sicherstellung der korrekten API-Kommunikation haben wir Swagger Codegen verwendet, um automatisch die notwendigen Klassen und Interfaces aus der OpenAPI-Definition zu generieren.

Die Angular HTTP-Client-Bibliothek wurde verwendet, um die API-Anfragen zu erstellen und die Daten zu empfangen.

Fehlerbehandlung

Die Fehlerbehandlung der API-Anfragen wurde mithilfe von Angulars `catchError` Operatoren umgesetzt. Bei einem Fehler in der Anfrage wird eine entsprechende Fehlermeldung an den Benutzer ausgegeben, um ein besseres Nutzererlebnis zu gewährleisten.

Datenmodell und Antwortstruktur

Die API liefert die Prozess-Effizienz-Daten in einem strukturierten Format zurück. Die Antwort enthält ein Array von Objekten, die jeweils die Prozess-Effizienz für einen bestimmten Zeitraum darstellen, einschliesslich der `id`, `startDate`, `endDate` und `processEfficiency` Werte.

Proxy

Um für die Entwicklung das Problem der CORS-Richtlinien zu umgehen und die Wartbarkeit zu erhöhen, haben wir einen Proxy verwendet. In der Datei `proxy.conf.json` wird die URL der API hinterlegt. Dadurch werden die Requests an die API aus den Komponenten an die spezifizierte URL weitergeleitet.

4.3 Testing

Dieses Kapitel beschreibt die Qualitätssicherung der Software durch Code- und Usability-Tests. Der Fokus liegt auf Unit Tests nach dem Prinzip des Test Driven Developments (TDD) mithilfe der Angular Testing Library. Zudem wird der Usability-Test erläutert, der unmoderiert mit einem Fragebogen durchgeführt wurde, um Feedback zur Benutzerfreundlichkeit zu erhalten.

4.3.1 Code Testing

Beim Testing haben wir uns so weit sinnvoll an den Ansatz des Test Driven Developments gehalten. Dabei haben wir uns auf Unit Tests konzentriert, da diese am effizientesten sind und die meisten Fehler abdecken. Für die Tests haben wir die Angular Testing Library verwendet.

Die Tests für die einzelnen Komponenten und Services haben wir in den jeweiligen `spec.ts` Files der Codebase hinzugefügt. Dabei haben wir uns auf die wichtigsten Funktionen und Methoden konzentriert, um die Testabdeckung zu maximieren.

Code Coverage

Das Abrufen der Code Coverage ist bereite standardmässig im entsprechenden Angluar Test-Command enthalten. Dabei wird die Code Coverage auf vier Ebenen angezeigt:

- **Statements**: Der Prozentsatz der Statements, die durch die Tests abgedeckt sind.
- **Branches**: Der Prozentsatz der Branches (wie zum Beispiel switch oder if Statements), die durch die Tests abgedeckt sind.
- **Functions**: Der Prozentsatz der Funktionen im Code, die durch die Tests abgedeckt sind.
- **Lines**: Der Prozentsatz der ausführbaren Codezeilen, die durch die Tests abgedeckt sind.

Die Code Coverage in Angular bietet standardmässig verschiedene Stufen an, die eine ungenügende, eine akzeptable und eine gute Code Coverage anzeigen. Wir haben uns das Ziel gesetzt, bei allen vier Ebenen eine Code Coverage der höchsten Stufe zu erreichen. Dabei sind wir jedoch den Kompromiss eingegangen, lediglich sinnvolle Tests zu integrieren, auch wenn das bedeutet, dass die Code Coverage lediglich im oberen Bereich der akzeptablen Stufe liegt.

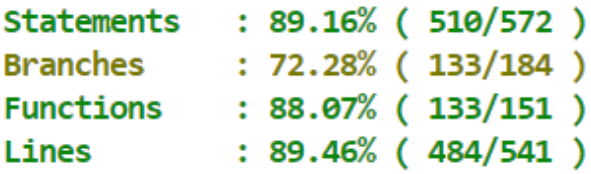


Abbildung 4.8: Code Coverage pro Ebene

Um die Code Coverage während der Entwicklung zu prüfen, haben wir den Angular Test-Command ausgeführt und anschliessend den generierten Report im Browser geöffnet. Dieser Report zeigt die Code Coverage auf den vier Ebenen an und gibt an, welche Teile des Codes nicht abgedeckt sind. Dabei wird direkt ersichtlich, welche Teile des Codes der Komponenten und Services noch nicht ausreichend getestet werden, was uns sehr dabei unterstützt hat zielgerichtete Tests zu schreiben.

4.3.2 Usability Testing

Die Durchführung eines Usability-Tests wurde auf den Zeitpunkt fixiert, an dem der MVP Stand erreicht wird und somit ein repräsentativer Prototyp zur Verfügung steht.

Vorbereitung

Während der Vorbereitung für den Usability-Test haben wir uns für einen unmoderierten Test entschieden. Das hatte den Grund, dass die Funktionalität des Tools einfach und intuitiv sein sollte und somit keine umfangreichen Erklärungen erfordert. Zudem waren individuelle Termine mit den Testpersonen aufgrund des Zeitmangels nicht realistisch möglich. Deshalb haben wir auf Basis eines Google Formulars einen Fragebogen erstellt, der die Testpersonen durch die Aufgaben führt und ihnen die Möglichkeit gibt, Feedback zu geben. Hierbei haben wir uns an die Guidelines des Website Usability Testings gehalten.

Der Fragebogen beinhaltete die folgenden Punkte:

- **Hinweis**: Einen Hinweis zum aktuellen Stand und zur Funktionalität des Tools.
- **Szenarios**: Eine Liste von Szenarios mit zugehörigen Aufgaben, die die Testpersonen durchführen sollen.
- **Feedback**: Möglichkeiten für die Testpersonen, für jedes Szenario Feedback zu geben.

- **Abschluss**: Einige abschliessende Fragen, um das allgemeine Feedback der Testpersonen zu erhalten.
- **Dank**: Ein Dankeschön für die Teilnahme am Usability-Test sowie die Möglichkeit, über die weitere Entwicklung des Tools informiert zu werden.
- **Hilfestellung**: Mehrere Screenshots des Tools mit Erklärungen zur Navigation und zu den Funktionen.

Um den Testpersonen Zugang zum Tool zu geben, haben wir einen neuen Branch erstellt, auf dem ein modifizierter Stand des Prototyps verfügbar war. Bei diesem Stand haben wir unter anderem sämtliche sensiblen Informationen wie URLs oder Access Tokens für die API entfernt, um die Sicherheit der Daten zu gewährleisten. Anschliessend haben wir diesem Branch mittels eines Bereitstellungstokens und einem entsprechenden Job in der CI/CD Pipeline in eine statische Webapp auf Azure deployed. Diese statische Webapp haben wir den Testpersonen zur Verfügung gestellt, damit sie die Aufgaben im Tool durchführen konnten. Um sicherzustellen, dass die Testpersonen nicht durch versehentliche Änderungen am Code und darauffolgendes Neudeployment des Tools verwirrt werden, haben wir die Ausführung des Deployment-Jobs auf manuell umgestellt.

Durchführung

Nach der Vorbereitung des Usability-Tests haben wir die statische Webapp basierend auf den Szenarien aus dem Fragebogen selbst ausprobiert, um sicherzustellen, dass die Aufgaben durchführbar sind und keine Fehler im Tool vorhanden sind. Anschliessend haben wir den Fragebogen sowie die URL der statischen Webapp an die Testpersonen verschickt und sie gebeten, den Test bis zum 17.12 durchzuführen, falls sie teilnehmen möchten.

Ergebnisse

Nach Ablauf der Frist hat sich herausgestellt, dass nur eine Testperson den Usability-Test durchgeführt hat. Deshalb sind die Ergebnisse des Usability-Tests nicht repräsentativ und können nicht als Grundlage für die Weiterentwicklung des Tools verwendet werden. Als Massnahme haben wir uns entschieden, direkt zu Beginn der Bachelorarbeit erneut einen Usability-Test durchzuführen, um sicherzustellen, dass das Tool benutzerfreundlich ist und die Anforderungen der Führungskräfte erfüllt. Dieser Test wir dann auch moderiert durchgeführt, um die Testpersonen bei Fragen und Problemen unterstützen zu können.

Dennoch lässt sich aus den Ergebnissen des Usability-Tests zumindest stichprobenartig ableiten, dass die Testperson keinerlei Probleme mit der Navigation und den Funktionen des Tools hatte. Das geht aus den Antworten auf die Frage wie einfach die Aufgabe zu erledigen war hervor, welche für jedes Szenario mit `sehr einfach` beantwortet wurde. Die Testperson hat auch hilfreiche Kritik geäussert. Beispielsweise wurde angemerkt, dass die aktuell verfügbaren Daten nicht ausreichen, um als Führungskraft Entscheidungen auf deren Basis zu treffen.

5 Reflexion

5.1 Ergebnisse

Im Rahmen der Arbeit konnten sämtliche funktionale und nicht-funktionale Anforderungen erfolgreich umgesetzt werden, die für den Minimum Viable Product (MVP)-Stand relevant waren. Im Folgenden wird dargelegt, welche Anforderungen erreicht wurden und welche nicht. Es wird ein konkreter Bezug zu den definierten Anforderungen hergestellt, ohne die Ergebnisse zu bewerten.

5.1.1 Erreichte Anforderungen

Funktionale Anforderungen

Die folgenden funktionalen Anforderungen wurden gemäss den festgelegten Kriterien erreicht:

- **UC1:** Erstellung eines personalisierten Dashboards *(MVP)*
  - Erfolgskriterien: Während dem Usability-Test war die Führungskraft dazu in der Lage, alle relevanten Informationen effizient zusammenzustellen.
- **UC4:** Historienverwaltung der Shopfloor-Pages *(MVP)*
  - Erfolgskriterien: Während dem Usability-Test war die Führungskraft dazu in der Lage, die Historie der Shopfloor-Pages anzuzeigen und zu vergleichen.

Nicht-funktionale Anforderungen

Die folgenden nicht-funktionalen Anforderungen wurden gemäss den festgelegten Kriterien erfüllt:

- **NFR1:** Intuitive Benutzeroberfläche *(MVP)*
  - Erfolgskriterien: Aus dem Usability-Test ging hervor, dass die Benutzeroberfläche intuitiv bedienbar ist.
- **NFR2:** Einfache Erweiterbarkeit um neue Widgets *(MVP)*
  - Erfolgskriterien: Neue Widgets können problemlos innerhalb eines Tages hinzugefügt werden. Dafür sorgt der modulare Aufbau der Applikation.
- **NFR6:** Main-Branch immer ausführbar *(MVP)*
  - Erfolgskriterien: Der Aufbau der CI/CD-Pipeline sorgt dafür, dass der Main-Branch immer ausführbar ist.

5.1.2 Nicht erreichte Anforderungen

Funktionale Anforderungen

Die folgenden funktionalen Anforderungen konnten im Rahmen der Arbeit nicht erreicht werden, da sie nicht im MVP enthalten waren:

- **UC2:** Präsentation der Shopfloor-Übersicht auf einem Bildschirm

  - Erfolgskriterien: Die Daten sind klar sichtbar und die Mitarbeitenden haben eine bessere Übersicht über die aktuellen Kennzahlen. Jedoch kann die Führungskraft keine Notizen und Skizzen zur Übersicht hinzufügen.

- **UC3:** Anpassung der Widgets
  - Erfolgskriterien: Die Widgets sind funktional, bieten jedoch noch keine Möglichkeit zur Anpassung der Inhalte.

- **UC5:** Benutzerverwaltung und Authentifizierung
  - Erfolgskriterien: Die Benutzerverwaltung und Authentifizierung sind noch nicht implementiert.

Nicht-funktionale Anforderungen

Die folgenden nicht-funktionalen Anforderungen konnten ebenfalls nicht erfüllt werden, da sie ausserhalb des MVP lagen:

- **NFR3:** Skalierbarkeit
  - Erfolgskriterien: Die Applikation ist noch nicht für eine hohe Anzahl von Benutzern ausgelegt, da es sich um einen Prototyp handelt.

- **NFR4:** Barrierefreiheit
  - Erfolgskriterien: Die Applikation ist nicht barrierefrei.

- **NFR5:** Authentifizierung
  - Erfolgskriterien: Die Authentifizierung ist noch nicht implementiert.

5.1.3 Rückmeldung vom Auftraggeber

Gemäss dem Feedback des Auftraggebers wurden die Erwartungen erfüllt, und die gelieferten Ergebnisse wurden positiv aufgenommen. Die Rückmeldung bestätigt die erfolgreiche Umsetzung der MVP-relevanten Anforderungen.

5.2 Schlussfolgerungen und Ausblick

5.2.1 Fazit der Umsetzung

Die Applikation befindet sich auf einem soliden Stand, mit einem erfolgreichen Proof of Concept, der eine gute Basis für die Weiterentwicklung bietet. Die bisherige Arbeit hat gezeigt, dass die Architektur und das Design sinnvoll und intuitiv sind.

Für den Prototyp lag der Fokus auf der Kernfunktionalität, weshalb einige weitergehende Features noch nicht umgesetzt wurden, aber in den Requirements bereits festgehalten wurden. Dazu zählen die Konfiguration von Widgets, Authentifizierung, die Speicherung im Backend und das Hinzufügen von Notizen in Shopfloors. Auch ist die Auswahl von Widgets aktuell sehr begrenzt, was nach einer Erweiterung der API seitens des Auftraggebers jedoch einfach zu erweitern sein sollte.

Die Codebasis bietet noch Potential für Verbesserungen. Es gibt teilweise Code-Duplikationen, die durch eine stärkere Abstrahierung von View-Komponenten reduziert werden könnten. Im Rahmen des Prototyps wurden ausserdem keine Massnahmen zur Performance-Optimierung

umgesetzt und es wurden keine Performance-Tests durchgeführt. In einer zukünftigen Version sollten Konzepte wie Lazy Loading, Code Splitting, On-Demand Data Loading und die Reduktion der Bundle-Grösse durch Tree Shaking umgesetzt werden, um die Performance zu optimieren.

Zudem ist der Wunsch aufgekommen, die Kennzahlen in den Widgets nach produzierten Produkten zu filtern. Dadurch würde die Akzeptanz der Applikation deutlich gesteigert und die Nutzung der Applikation gefördert. Allerdings müsste mit dem Auftraggeber abgeklärt werden, ob die API entsprechend erweitert werden kann.

Nach der Implementierung der Verbesserungen ist ein erneuter Usability-Test mit den Führungskräften erforderlich, da der in der Studienarbeit durchgeführte Usability-Test nicht repräsentativ ist. So kann sichergestellt werden, dass die Anwendung den Anforderungen der Nutzer entspricht und eine hohe Benutzerakzeptanz erreicht werden kann.

5.2.2 Fazit des Projektmanagements

Rückblickend hätte eine intensivere Planungsphase im Vorfeld dazu beitragen können, den Refactoring-Aufwand zu reduzieren.

Da die Aufgabenstellung zum Anfang der Studienarbeit offen war, wurde viel Zeit in die Analyse und User-Research-Phase investiert. Dies hatte zur Folge, dass sich die Implementierung nach hinten verschob und schliesslich nur wenig Zeit für die eigentliche Umsetzung des Prototyps zur Verfügung stand. Dennoch hat die intensive Auseinandersetzung mit der Problemstellung die Entwicklung insgesamt positiv geprägt, da fundierte Entscheidungen getroffen werden konnten. Die Evaluation der Idee zum digitalen Shopfloor Management wurde durchweg positiv aufgenommen. Dies bestätigte die Relevanz der Thematik, die auch beim Auftraggeber bereits länger als Diskussionspunkt bestand. Die positive Resonanz verdeutlicht, dass die eingeschlagene Richtung viel Potenzial bietet.

Eine weitere Herausforderung war der Bedarf an spontanem Feedback durch den Auftraggeber. Da keine längerfristig geplanten Feedbackschleifen definiert wurden, war es erforderlich, Meetings mit dem Auftraggeber kurzfristig zu organisieren. Dies stellte zeitweise eine organisatorische Hürde dar. Für zukünftige Projekte sollte hierauf ein stärkeres Augenmerk gelegt werden, um eine strukturiertere Zusammenarbeit zu gewährleisten. Eine frühzeitige und detaillierte Planung der Feedbackzyklen könnte dazu beitragen, die Kommunikation effizienter zu gestalten.

Aufgrund organisatorischer Gründe, wie Zeitdruck und der fehlenden Möglichkeit zur Terminierung zwischen der Fertigstellung des MVP und der Abgabe der Arbeit, wurde ein unmoderierter Usability-Test durchgeführt. Dieser Test wurde jedoch nur von einer von sechs Personen beantwortet. Eine Kontaktierung der Führungspersonen über den Projektpartner hätte möglicherweise zu einer höheren Rücklaufquote geführt. Unmoderierte Usability-Tests erfordern in der Regel eine grössere Nutzerbasis, um aussagekräftige Ergebnisse zu erzielen. Zukünftige Usability-Tests sollten entweder moderiert durchgeführt oder mit einer deutlich grösseren Anzahl an Teilnehmenden umgesetzt werden. Der erstellte Fragenkatalog sowie das aufgesetzte Deployment bieten jedoch eine gute Basis für zukünftige Tests.

5.2.3 Ausblick

Es ist geplant, die Studienarbeit im Rahmen einer Bachelorarbeit weiterzuführen, um den entstandenen Prototyp umzusetzen und die in dieser Arbeit behandelten Themen weiter zu vertiefen. Neben den bereits erwähnten Verbesserungen gibt es eine Reihe weiterer Funktionen, die sich für diese Weiterführung eignen könnten.

Eine Möglichkeit wäre die Implementierung einer neuen Seite zur Verwaltung von Aufgaben

(Tasks), die aus dem wöchentlichen Shopfloor entstehen. Diese Erweiterung würde es den Nutzern ermöglichen, Aufgaben effizienter zu organisieren und den Überblick über ihre Arbeitsabläufe zu behalten. Um sicherzustellen, dass diese Funktion den tatsächlichen Bedürfnissen der Nutzer entspricht, ist eine weitere Untersuchung der Benutzeranforderungen erforderlich. Dabei könnte es notwendig sein, den User-Research in einem grösseren Rahmen durchzuführen, der nicht nur die bisherigen Nutzergruppen umfasst, sondern auch Personen aus anderen Abteilungen wie dem Support, der Instandhaltung und der Prozessverbesserung einbezieht. Die Einbeziehung dieser Abteilungen könnten wertvolle Perspektiven bieten, und eine firmenweite Nutzung der Applikation zu fördern.

Zusätzlich zu dieser Funktionalität könnte die Anwendung in Zukunft auch um weitere Features erweitert werden, die den Workflow der Nutzer weiter optimieren und die Anwendung noch benutzerfreundlicher gestalten. Diese Features könnten beispielsweise erweiterte Suchfunktionen, Benachrichtigungsmechanismen oder die Integration mit anderen Systemen umfassen, um den Arbeitsalltag der Nutzer noch effizienter zu gestalten.

Literaturverzeichnis

[1] Nielsen Norman Group. (2020). Contextual inquiry. Retrieved October 4, 2024, from https://www.nngroup.com/articles/contextual-inquiry/

[2] Nielsen Norman Group. (2020). Task analysis. Retrieved October 4, 2024, from https://www.nngroup.com/articles/task-analysis/

[3] Nielsen Norman Group. (2024). Affinity diagram. Retrieved October 4, 2024, from https://www.nngroup.com/articles/affinity-diagram/

[4] Miro. (2024). Was ist ein Wireframe? Retrieved November 09, 2024, from https://miro.com/de/wireframing/was-ist-wireframing/

[5] Nielsen Norman Group. (2015). Personas Make Users Memorable for Product Team Members. Retrieved October 30, 2024, from https://www.nngroup.com/articles/persona/

[6] Nielsen Norman Group. (2019). Unmoderated User Tests. Retrieved December 09, 2024, from https://www.nngroup.com/articles/unmoderated-usability-testing/

[7] Jansen, O. (2024). Contextual inquiry. Retrieved October 10, 2024, from https://odette-jansen.notion.site/Contextual-Inquiry-b1ff7d718b234a01a20a77bcf8a3bf4b/

[8] BrowserStack. (2024). Website Usability Testing. Retrieved December 09, 2024, from https://www.browserstack.com/guide/website-usability-testing

[9] Scherrer, A. (2021). *Kennzahlen Fact Sheet* [Internal Document, SFS Group AG].

[10] Scherrer, A. (2023). *Schulung Eolos Werkzeuge* [Internal Document, SFS Group AG].

[11] Marketing and Corporate Communications (2024). *Corporate Design Manual* [Internal Document, SFS Group AG].

[12] Almaz, A., and Zimmermann, U. (2022). *Nutzerzentrierte Entwicklung eines digitalen Bestellprozesses für Rohmaterialien* [Bachelor thesis, OST - Ostschweizer Fachhochschule].

[13] This Person Does Not Exist. (2023). *This Person Does Not Exist*. Retrieved October 30, 2024, from https://this-person-does-not-exist.com/en

Abbildungsverzeichnis

Tabellenverzeichnis

A Sofware-Dokumentation

A.1 Anforderungen

Nach der Analyse und der Auswahl des Themas wurden die Anforderungen an das Shopfloor-Tool für Führungskräfte definiert. Diese Anforderungen sind in funktionale und nichtfunktionale Anforderungen unterteilt. Die funktionalen Anforderungen beschreiben, welche Funktionen das Tool bereitstellen muss, um den Anforderungen der Führungskräfte gerecht zu werden. Die nichtfunktionalen Anforderungen beschreiben zusätzliche Anforderungen, die unabhängig vom Funktionsumfang des Tools sind. Für jede Anforderung ist zudem definiert, ob die jeweilige Anforderung für den MVP oder für eine spätere Version des Tools umgesetzt werden muss. Die verwendeten Templates für die Anforderungen sind leicht modifizierte Versionen der Vorlagen aus den Übungen des Software Engineering 2 Moduls.

A.1.1 Funktionale Anforderungen

UC1	**Erstellung eines personalisierten Dashboards**
Szenario	- Die Führungskraft erstellt ein personalisiertes Dashboard durch Auswahl von Widgets. - Die Führungskraft kann relevante KPIs wie wöchentliche Trends der Prozesseffizienz oder Ausschussentwicklung hinzufügen und anordnen.
Vorbedingungen	Die Führungskraft ist authentifiziert und hat Zugriff auf das Tool.
Nachbedingungen	Ein gespeichertes, personalisiertes Dashboard steht bereit zur Anzeige.
Akteure	Führungskräfte
Erfolgskriterien	Die Führungskraft kann alle relevanten Informationen effizient zusammenstellen.

Tabelle A.1: funktionale Anforderung 1

UC2	Präsentation der Shopfloor-Übersicht auf einem Bildschirm
Szenario	- Die Führungskraft zeigt die personalisierte Shopfloor-Übersicht auf einem grossen Bildschirm zur Besprechung mit den Mitarbeitenden an. - Die Führungskraft kann während dem Shopfloor Notizen und Skizzen zur Übersicht hinzufügen. - Die Eingabe ist für Maus und Tastatur sowie Touchscreen optimiert.
Vorbedingungen	Die Führungskraft hat eine Übersicht erstellt und Zugang zu einem verbundenen Bildschirm.
Nachbedingungen	Die Übersicht wird für alle sichtbar auf dem Bildschirm angezeigt.
Akteure	Führungskräfte
Erfolgskriterien	Die Daten sind klar sichtbar, die Führungskraft kann Ergänzungen hinzufügen und die Mitarbeitenden haben eine bessere Übersicht über die aktuellen Kennzahlen.

Tabelle A.2: funktionale Anforderung 2

UC3	Anpassung der Widgets
Szenario	- Die Führungskraft kann Widgets anpassen, indem sie die gewünschten Inhalte konfiguriert. - Widgets können für unterschiedliche Zeiträume angepasst werden (Datumsauswahl).
Vorbedingungen	Die Führungskraft hat Zugriff auf das Tool und die vorhandenen Widgets.
Nachbedingungen	Die konfigurierten Widgets werden auf dem Dashboard aktualisiert.
Akteure	Führungskräfte
Erfolgskriterien	Die Widgets sind funktional und bieten nach der Konfiguration aktuelle, relevante Daten.

Tabelle A.3: funktionale Anforderung 3

UC4	Historienverwaltung der Shopfloor-Pages
Szenario	- Die Führungskraft ruft vergangene Shopfloor-Pages auf. - Ein automatischer Datenabruf stellt sicher, dass die Daten der entsprechenden Woche angezeigt werden.
Vorbedingungen	Vorhandensein der gespeicherten Shopfloor-Pages im Backend oder im Browserstorage.
Nachbedingungen	Historische Daten sind abrufbar und können dargestellt werden.
Akteure	Führungskräfte
Erfolgskriterien	Die Führungskraft kann die Historie der Shopfloor-Pages anzeigen und vergleichen.

Tabelle A.4: funktionale Anforderung 4

UC5	Benutzerverwaltung und Authentifizierung
Szenario	- Die Führungskraft meldet sich an und authentifiziert sich, um Zugriff auf das Tool zu erhalten. - Die Authentifizierung erfolgt analog jener von bestehenden Systemen.
Vorbedingungen	Die Führungskraft hat ein gültiges Konto im Authentifizierungssystem.
Nachbedingungen	Die Führungskraft hat Zugriff auf personalisierte Daten und Einstellungen.
Akteure	Führungskräfte, Systemadministratoren
Erfolgskriterien	Der Zugang ist sicher und für berechtigte Führungskräfte problemlos möglich.

Tabelle A.5: funktionale Anforderung 5

A.1.2 Nicht-funktionale Anforderungen

ID	NFR1
Beschreibung	Die Benutzeroberfläche muss intuitiv bedienbar sein, damit die Führungskräfte ohne Schulung die wichtigsten Funktionen verstehen.
Anforderung	Widgets und Layouts sollen übersichtlich und einfach zu konfigurieren sein.
Messbarkeit	Feedback von Führungskräften (mindestens 80% Zufriedenheit in internen Tests).
Priorität	Hoch
Deadline	MVP-Version

Tabelle A.6: nichtfunktionale Anforderung 1

ID	**NFR2**
Beschreibung	Die Lösung soll einfach um neue Widgets und Datenquellen erweiterbar sein.
Anforderung	Ein modulares System zur einfachen Integration neuer Widgets.
Messbarkeit	Implementierungsdauer neuer Widgets maximal 1 Tag.
Priorität	Mittel
Deadline	Nach der MVP-Phase

Tabelle A.7: nichtfunktionale Anforderung 2

ID	**NFR3**
Beschreibung	Die Software sollte skalierbar sein, um bei wachsender Zahl von Benutzenden stabil zu bleiben.
Anforderung	Unterstützung von mindestens 200 gleichzeitigen Benutzenden ohne Performanceeinbussen.
Messbarkeit	Durchführung von Lasttests zur Überprüfung der Performance.
Priorität	Mittel
Deadline	Vor Vollausbau

Tabelle A.8: nichtfunktionale Anforderung 3

ID	**NFR4**
Beschreibung	Das Tool sollte barrierefrei gestaltet sein, um von allen Personen, auch mit Einschränkungen, verwendet werden zu können.
Anforderung	Kompatibilität mit Screenreadern und vollständige Tastatursteuerung.
Messbarkeit	WCAG 2.1-Konformitätstests durchführen.
Priorität	Mittel
Deadline	Vor Einführung der ersten vollständigen Version

Tabelle A.9: nichtfunktionale Anforderung 4

ID	**NFR5**
Beschreibung	Die Anwendung muss sicher sein und sicherstellen, dass nur autorisierte Führungskräfte auf Daten zugreifen.
Anforderung	Verschlüsselung ruhender Daten.
Messbarkeit	Durchführung von Penetrationstests und Code-Reviews.
Priorität	Hoch
Deadline	Nach der MVP-Phase

Tabelle A.10: nichtfunktionale Anforderung 5

ID	**NFR6**
Beschreibung	Der aktuelle Stand der Anwendung kann jederzeit ausgeführt werden.
Anforderung	Der Codestand im Main-Branch ist jederzeit fehlerfrei ausführbar.
Messbarkeit	Die Pipeline läuft ohne Fehler.
Priorität	Hoch
Deadline	Vor MVP-Launch

Tabelle A.11: nichtfunktionale Anforderung 6

B Projektmanagement

Dieses Kapitel beschreibt das Projektmanagement für unser Projekt.

B.1 SCRUM

Das Projekt wird mit der Scrum-Methode durchgeführt, die auf der agilen Softwareentwicklung basiert. In Scrum werden die Arbeiten in zweiwöchige Sprints unterteilt, in denen bestimmte Aufgaben erledigt werden. Zu Beginn jedes Sprints findet ein Scrum-Meeting statt, um den Fortschritt zu besprechen und die nächsten Schritte zu planen.

B.1.1 Sprints

- Arbeitstage: Donnerstag und Freitag
- Dauer: 2 Wochen
- Scrum-Meeting: Am ersten Donnerstag des Sprints

B.2 Beratungsgespräche

Die Beratungsgespräche finden wöchentlich statt. In diesen Gesprächen werden die Fortschritte besprochen und offene Fragen geklärt. Die Beratungsgespräche dienen auch dazu, Probleme zu identifizieren und Lösungen zu finden. Die Beratungsgespräche finden jeden Donnerstag um 13:30 Uhr per Teams statt und dauern maximal eine Stunde. Teilnehmer sind die Projektmitglieder und der Betreuer. Die Projektmitglieder führen zu jedem Beratungsgespräch ein Protokoll, welche der Dokumentation hinzugefügt werden.

B.3 Meilensteine

Die Meilensteine für das Projekt sind wie in untenstehender Tabelle B.1 definiert:

Meilenstein	Datum
Anforderungsanalyse	27.10.2024
Entwurf	10.11.2024
MVP und Implementation	08.12.2024
Testing	15.12.2024
Abagbe	20.12.2024

Tabelle B.1: Meilensteine und ihre Deadlines

B.4 Arbeitspakete

Die Arbeitspakete für das Projekt umfassen die Punkte aus untenstehender Tabelle B.2:

Kategorie	Arbeitspakete
Planung	- Projektplan - SCRUM Meetings
Anforderungsanalyse	- Requirements - User Research - Umsystemanalyse
Entwurf	- MVP Definition - UI Sketch - Architektur Research - Persona Definition - Risikoanalyse
Implementation	- Refactoring und Verbesserungen - Entwicklung Frontend - Infrastruktur - Entwicklung Frontend MVP
Testing	- Usability Tests
Dokumentation	- Dokumentation Template - Dokumentation verfassen - Dokumentation abschliessen
Meeting	- Client Meeting - Beratungsgespräche

Tabelle B.2: Kategorien und zugehörige Arbeitspakete

B.5 Dokumentation und Nachverfolgung

Die Planung und der Fortschritt des Projekts werden regelmässig dokumentiert. Die Arbeitszeiten werden auf der Ebene der Arbeitspakete erfasst, um eine genaue Nachverfolgung und Transparenz zu gewährleisten.

B.6 Entwicklungswerkzeuge und eingesetzte Software

Tools und Software	Verwendung
Cypress und Jest	Frontend Testing
ESLint	Frontend Linting
GitLab	Code-Versionierung, CI, CD, Issue-Management, Sprint-Planung
Miro	Affinity Diagramming, Personas, Wireframes
MS Office Produkte	Dokumentation
Clockify	Zeiterfassung

Tabelle B.3: Entwicklungswerkzeuge

B.7 Risiken

B.7.1 Risikoanalyse

Wir haben eine Risikoanalyse durchgeführt, um mögliche Risiken zu identifizieren, zu bewerten und gegebenenfalls zu vermindern. Zuerst wurden die relevantesten Risiken identifiziert, die im Rahmen des Projekts auftreten könnten. Diese wurden dann nach ihrem Schadenmass und ihrer Eintrittswahrscheinlichkeit bewertet. Daraus ergab sich für jedes Risiko eine Risikostufe. In einem zweiten Schritt haben wir Massnahmen definiert, um die Risiken abzuschwächen oder falls möglich sogar zu minimieren. Danach wurden die Risiken anhand der oben genannten Faktoren erneut bewertet und die Risikostufe neu bestimmt.

		PRE-MITIGATION			ABSCHWÄCHUNGEN / WARNUNGEN / ABHILFEMASSNAHMEN	NACH DER RISIKOMINDERUNG			
ID	RISIKO	SCHADENMASS	WAHRSCHEINLICHKEIT	RISIKOSTUFE		SCHADENMASS	WAHRSCHEINLICHKEIT	RISIKOSTUFE	AKZEPTABEL, UM FORTZUFAHREN?
R1	Requirement Änderungen	hoch	möglich	**MITTEL**	solide MVP Definition, modulares Design, iterative Entwicklung	mittel	unwahrscheinlich	**MITTEL**	**JA**
R2	Mangelnde Akzeptanz durch Teamleiter	sehr hoch	möglich	**HOCH**	User-centered Design, Orientierung an Inquiry-Auswertung, UI an bestehende Tools anlehnen	mittel	unwahrscheinlich	**MITTEL**	**JA**
R3	Erforderliche Drittpersonen nicht erreichbar	hoch	wahrscheinlich	**HOCH**	Frühzeitige Anfragen an Drittpersonen, Projektplan aktuell halten, Regelmässige Terminvorschau in SCRUM-Meetings, regelmässiger Austausch mit Drittpersonen	mittel	möglich	**MITTEL**	**JA**
R4	Zeitplan kann nicht eingehalten werden	hoch	wahrscheinlich	**HOCH**	regelmässige SCRUM-Meetings und Beratungsgespräche, MVP minimal und variabel halten, regelmässiger Austausch mit Drittpersonen	mittel	möglich	**MITTEL**	**JA**

Abbildung B.1: Risikoanalyse

B.8 Risikomatrix

In einem letzten Schritt haben wir die beiden Risikostufen, welche sich aus jedem Risko ergeben haben in die Risikomatrix eingetragen. Aus der so entstandenen Matrix war nun ablesbar, ob die Risiken nach der Mitigierung unter der Akzeptanzlinie liegen. Da dies bei allen Risiken der Fall war, lag die Schlussfolgerung nahe, dass die Risiken für das Projekt akzeptabel sind und mit dem Projekt fortgefahren werden kann.

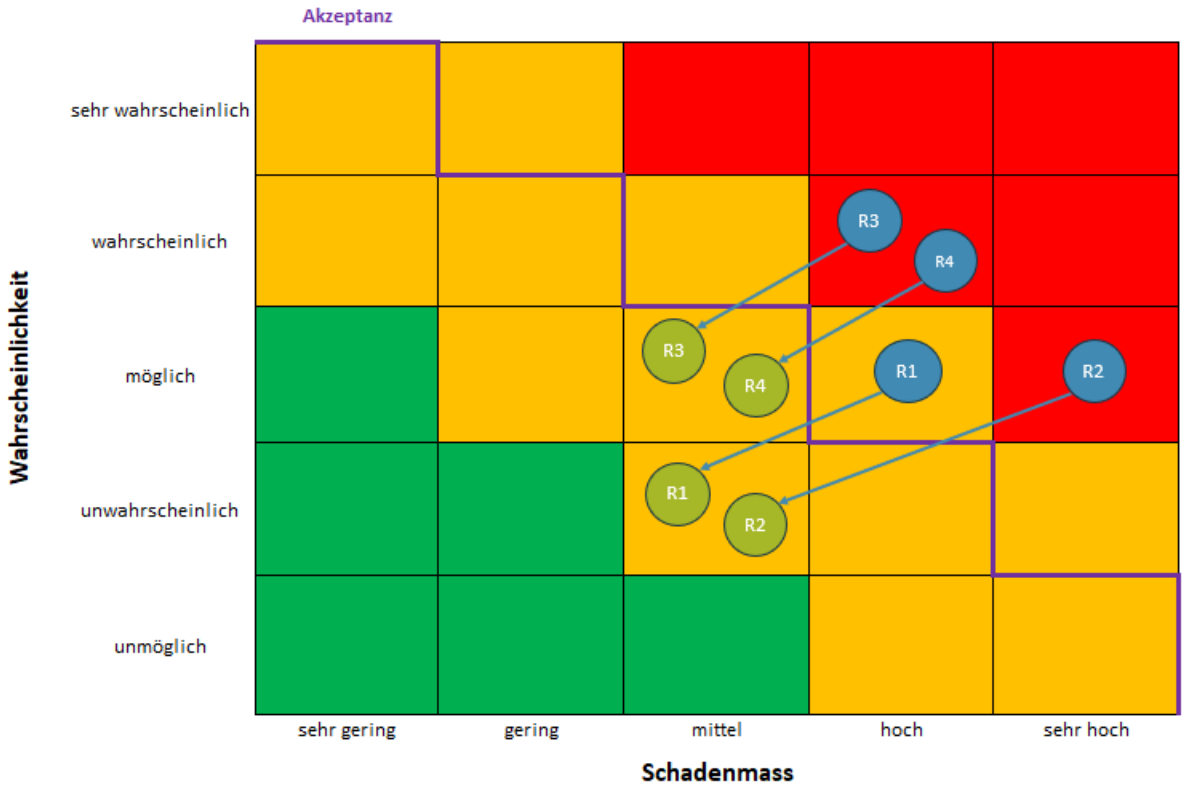


Abbildung B.2: Risikoanalyse

Natürlich ist die Risikoanalyse ein kontinuierlicher Prozess und wird regelmässig durchgeführt, um mögliche Risiken frühzeitig zu erkennen und zu minimieren.

C Projektplan

C.1 SOLL-Übersicht

SA Project Plan		W1 16.09.24	W2 23.09.24	W3 30.09.24	W4 07.10.24	W5 14.10.24	W6 21.10.24	W7 28.10.24	W8 04.11.24	W9 11.11.24	W10 18.11.24	W11 25.11.24	W12 02.12.24	W13 09.12.24	W14 16.12.24
Arbeitspaket	**Zeit SOLL**	**Zero Sprint**		**Iteration 1**		**Iteration 2**		**Iteration 3**		**Iteration 4**		**Iteration 5**		**Iteration 6**	
Planung	26														Abgabe
Projektplan	14														
SCRUM Meetings	12														
Anforderungsanalyse	70				Anforderungsanalyse										
Requirements	14														
User Research	48														
Umsystemanalyse	8														
Entwurf	56								Entwurf						
MVP Definition	8														
UI Sketch	16														
Architektur Research	16														
Persona Definition	8														
Risikoanalyse	8														
Implementation	170										MVP & Implementation				
Refactoring und Verbesserungen	16														
Entwicklung Frontend	18														
Infrastruktur	24														
Entwicklung Frontend MVP	112														
Testing	16													Testing	
Usability Tests	16														
Dokumentation	104														
Dokumentation Template	16														
Dokumentation verfassen	64														
Dokumentation abschliessen	24														
Meeting	38														
Client Meeting	10														
Beratungsgespräche	28														
	480														

Abbildung C.1: Projektplan SOLL-Übersicht

C.2 IST-Übersicht

SA Project Plan				W1 16.09.24	W2 23.09.24	W3 30.09.24	W4 07.10.24	W5 14.10.24	W6 21.10.24	W7 28.10.24	W8 04.11.24	W9 11.11.24	W10 18.11.24	W11 25.11.24	W12 02.12.24	W13 09.12.24	W14 16.12.24
Arbeitspaket	**Zeit SOLL**	**Zeit IST**		**Zero Sprint**		**Iteration 1**		**Iteration 2**		**Iteration 3**		**Iteration 4**		**Iteration 5**		**Iteration 6**	
Planung	26	31.5	121%														Abgabe
Projektplan	14	14.5	104%														
SCRUM Meetings	12	17	142%														
Anforderungsanalyse	70	70	100%					Anforderungsanalyse									
Requirements	14	9	64%														
User Research	48	56	117%														
Umsystemanalyse	8	5	63%														
Entwurf	56	49.5	88%									Entwurf					
MVP Definition	8	6.5	81%														
UI Sketch	16	22	138%														
Architektur Research	16	8.5	53%														
Persona Definition	8	6	75%														
Risikoanalyse	8	6.5	81%														
Implementation	170	192.5	113%											MVP & Implementation			
Refactoring und Verbesserungen	16	21	131%														
Entwicklung Frontend	18	10.25	57%														
Infrastruktur	24	19.5	81%														
Entwicklung Frontend MVP	112	141.75	127%														
Testing	16	13	81%													Testing	
Usability Tests	16	13	81%														
Dokumentation	104	122.5	118%														
Dokumentation Template	16	19.25	120%														
Dokumentation verfassen	64	69.75	109%														
Dokumentation abschliessen	24	33.5	140%														
Meeting	38	34	89%														
Client Meeting	10	9	90%														
Beratungsgespräche	28	34	121%														
	480	513	107%														

Abbildung C.2: Projektplan IST-Übersicht

D Projektmonitoring

D.1 Aufwandverteilung

Aufwandverteilung Planung

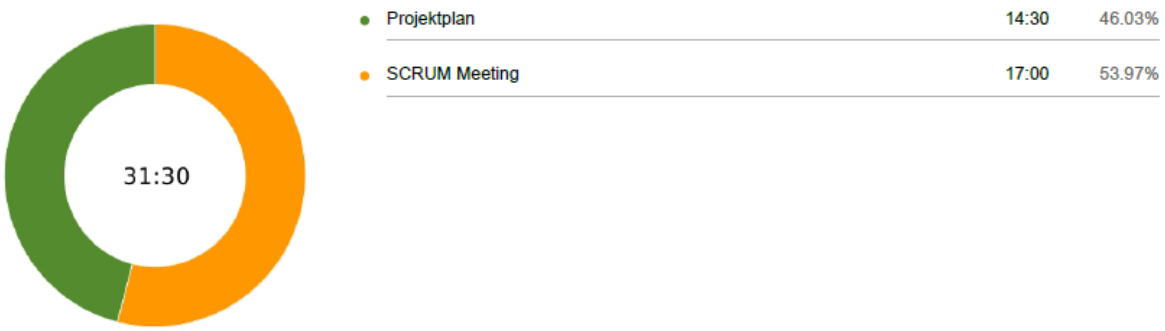


Abbildung D.1: Aufwandverteilung Planung

Aufwandverteilung Anforderungsanalyse

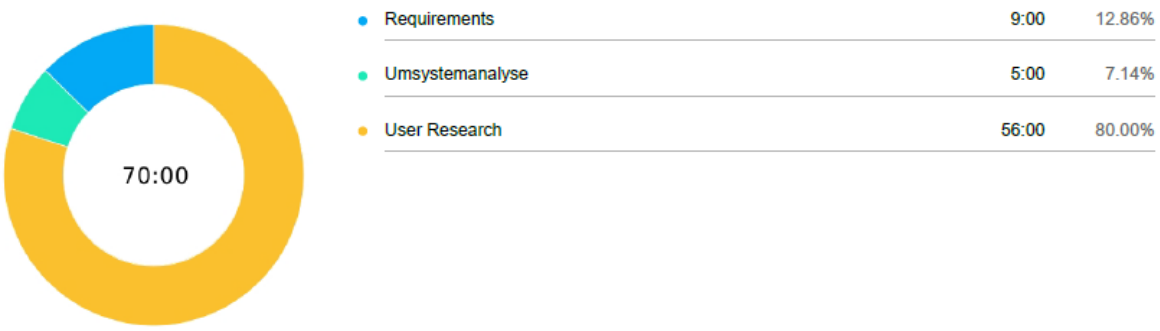


Abbildung D.2: Aufwandverteilung Anforderungsanalyse

Aufwandverteilung Entwurf

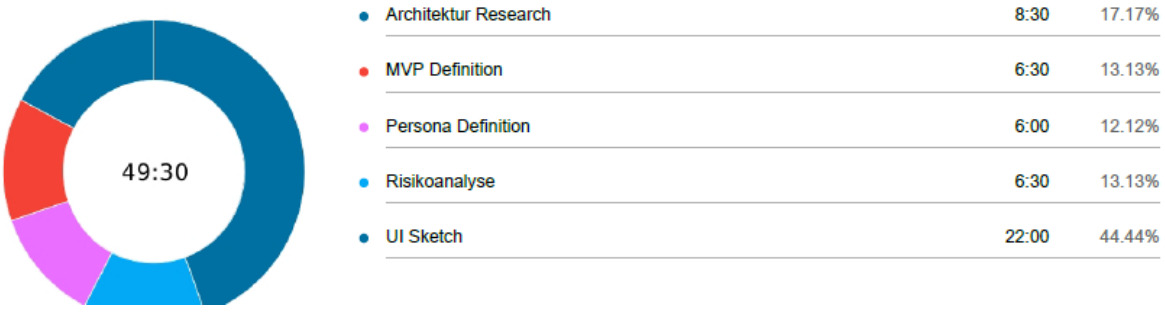


Abbildung D.3: Aufwandverteilung Entwurf

Aufwandverteilung Implementation

Abbildung D.4: Aufwandverteilung Implementation

Aufwandverteilung Testing

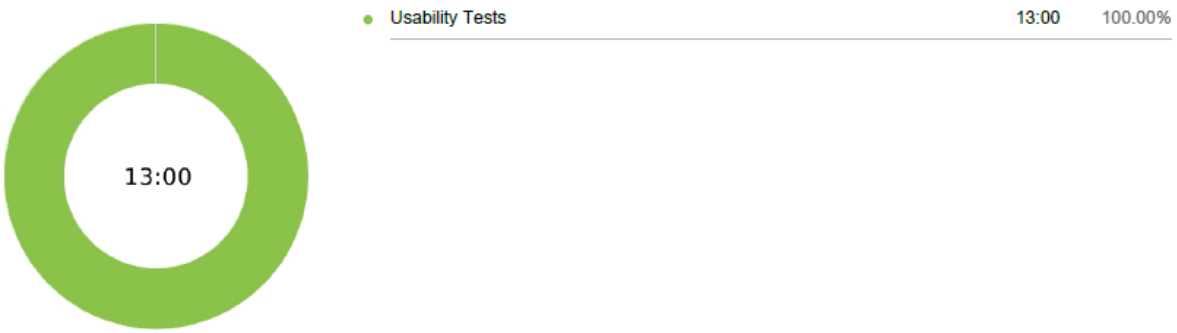


Abbildung D.5: Aufwandverteilung Testing

Aufwandverteilung Dokumentation

Abbildung D.6: Aufwandverteilung Dokumentation

Aufwandverteilung Meeting

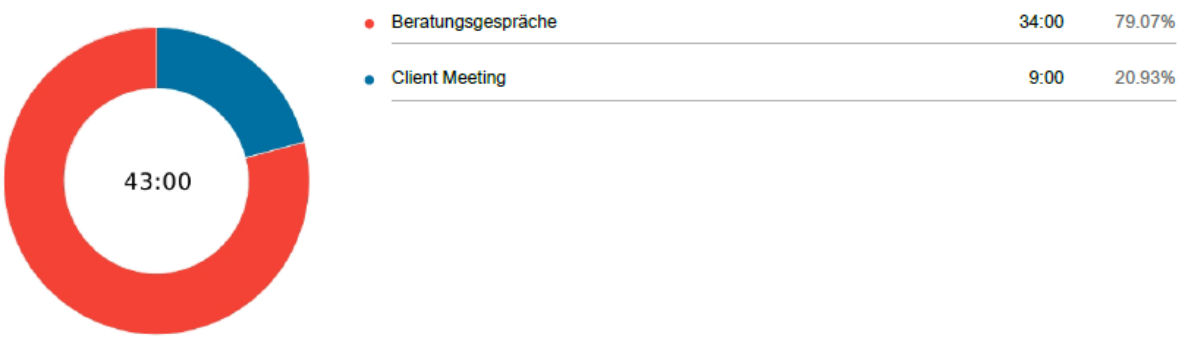


Abbildung D.7: Aufwandverteilung Meeting

D.2 Code-Metriken

Untenstehend sind die Code-Metriken für den Source-Code aufgeführt.

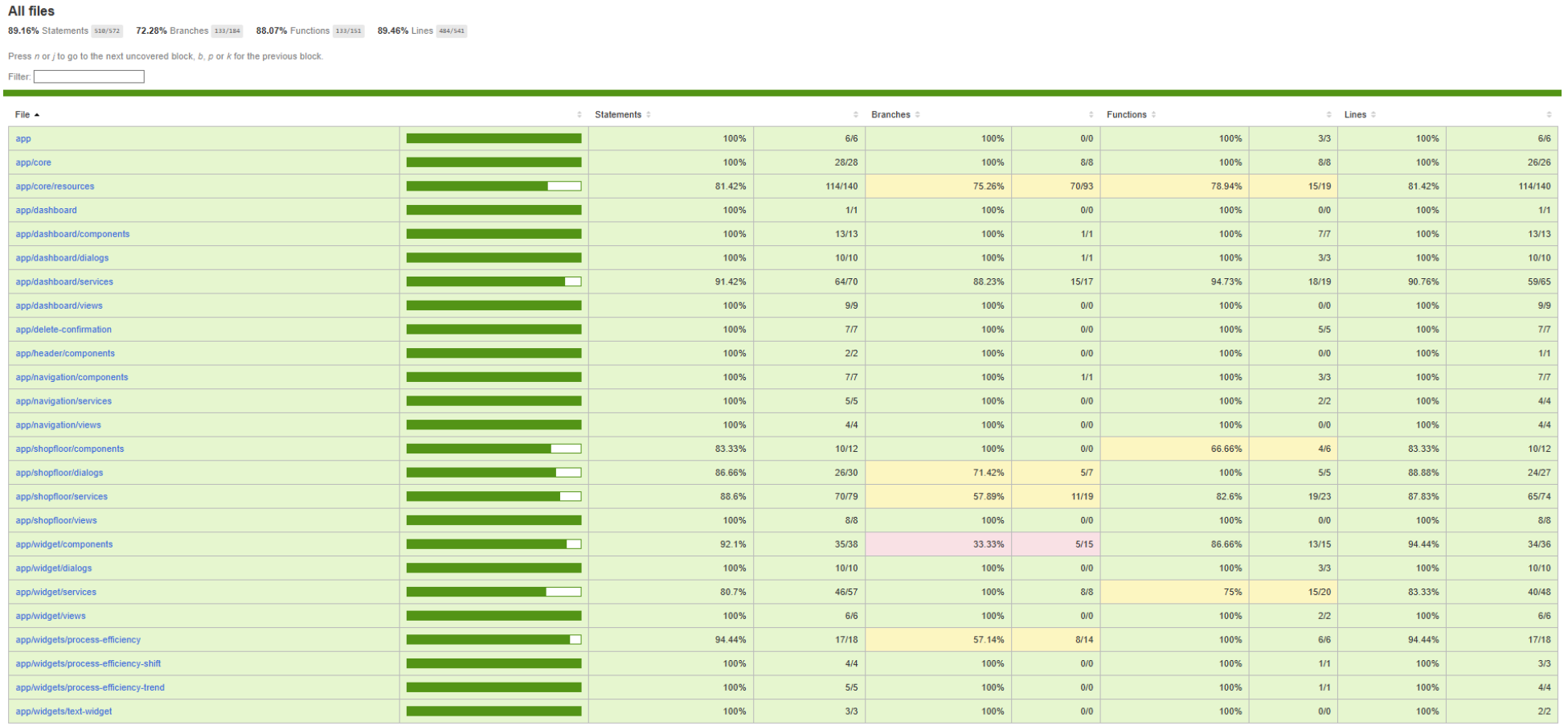
Codezeilen

Die Metriken zu den Codezeilen wurden mit dem Command Line Tool `cloc` erstellt. Die Werte zeigen die Anzahl Zeilen ohne Kommentare und Leerzeichen.

Sprache	Gesamt	Davon Tests
TypeScript	4348	2268
SCSS	295	0
HTML	119	0

Tabelle D.1: Codemetriken

Code Coverage Report



All files

89.16% Statements 510/572 **72.28%** Branches 133/184 **88.07%** Functions 133/151 **89.46%** Lines 484/541

Press *n* or *j* to go to the next uncovered block, *b*, *p* or *k* for the previous block.

Filter:

File	Statements		Branches		Functions		Lines	
app	100%	6/6	100%	0/0	100%	3/3	100%	6/6
app/core	100%	28/28	100%	8/8	100%	8/8	100%	26/26
app/core/resources	81.42%	114/140	75.26%	70/93	78.94%	15/19	81.42%	114/140
app/dashboard	100%	1/1	100%	0/0	100%	0/0	100%	1/1
app/dashboard/components	100%	13/13	100%	1/1	100%	7/7	100%	13/13
app/dashboard/dialogs	100%	10/10	100%	1/1	100%	3/3	100%	10/10
app/dashboard/services	91.42%	64/70	88.23%	15/17	94.73%	18/19	90.76%	59/65
app/dashboard/views	100%	9/9	100%	0/0	100%	0/0	100%	9/9
app/delete-confirmation	100%	7/7	100%	0/0	100%	5/5	100%	7/7
app/header/components	100%	2/2	100%	0/0	100%	0/0	100%	1/1
app/navigation/components	100%	7/7	100%	1/1	100%	3/3	100%	7/7
app/navigation/services	100%	5/5	100%	0/0	100%	2/2	100%	4/4
app/navigation/views	100%	4/4	100%	0/0	100%	0/0	100%	4/4
app/shopfloor/components	83.33%	10/12	100%	0/0	66.66%	4/6	83.33%	10/12
app/shopfloor/dialogs	86.66%	26/30	71.42%	5/7	100%	5/5	88.88%	24/27
app/shopfloor/services	88.6%	70/79	57.89%	11/19	82.6%	19/23	87.83%	65/74
app/shopfloor/views	100%	8/8	100%	0/0	100%	0/0	100%	8/8
app/widget/components	92.1%	35/38	33.33%	5/15	86.66%	13/15	94.44%	34/36
app/widget/dialogs	100%	10/10	100%	0/0	100%	3/3	100%	10/10
app/widget/services	80.7%	46/57	100%	8/8	75%	15/20	83.33%	40/48
app/widget/views	100%	6/6	100%	0/0	100%	2/2	100%	6/6
app/widgets/process-efficiency	94.44%	17/18	57.14%	8/14	100%	6/6	94.44%	17/18
app/widgets/process-efficiency-shift	100%	4/4	100%	0/0	100%	1/1	100%	3/3
app/widgets/process-efficiency-trend	100%	5/5	100%	0/0	100%	1/1	100%	4/4
app/widgets/text-widget	100%	3/3	100%	0/0	100%	0/0	100%	2/2

Abbildung D.8: Code Coverage Report